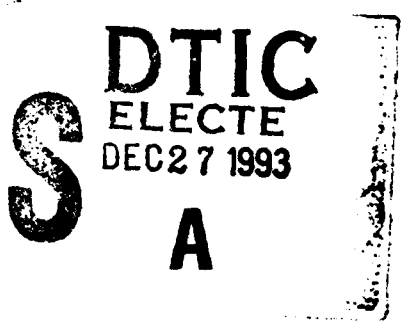
Computer Simulation and Experiments on the Quasi-static Mechanics and Transport Properties of Granular Materials

by



Xuejin Zhuang and J. D. Goddard

Department of Applied Mechanics
and Engineering Sciences
University of California, San Diego
9500 Gilman Drive
La Jolla, CA 92093-0310

Research Report GR 93-01

1 October 1993

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	1 October 1993	10/01/92 - 09/30/93 ANNUAL

4. TITLE AND SUBTITLE
Computer Simulation and Experiments on the Quasi-static Mechanics and Transport Properties of Granular Materials

5. FUNDING NUMBERS
Project No. 2302
Task No. CS

6. AUTHOR(S)
Xuejin Zhuang and J.D. Goddard (Principal Investigator)

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of California, San Diego
9500 Gilman Drive
La Jolla, CA 92093-0310

8. PERFORMING ORGANIZATION REPORT NUMBER
AFOSR
F49620-92-J-0037

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR[redacted] NA
AFOSR/NA
Building 410
Bolling AFB, DC 20332-6448

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFOSR
F49620-92-J-0037

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This report presents the results of a new quasi-static algorithm developed to compute the mechanical and scalar transport properties of three-dimensional sphere assemblages. The algorithm incorporates several new techniques, including a shuffling algorithm for generation of initial random granular packings, an improved microcell-adjacency method to accelerate particle-contact search, and a relaxation method to overcome a singularity in the quasi-linear system of equilibrium equations.

The calculated Reynolds dilatancy for random dense-packed granular assemblages is found to depend on the interparticle friction, contrary to Reynolds' original hypothesis, and the use of linear contact mechanics is found to be valid near the ideal rigid-particle limit.

Triaxial compression tests, employing steel balls as electrically conductive granular particles, confirm our simulation of both mechanical and transport properties, when account is taken of the actual electrical contact resistance between steel balls. The latter is much higher than for Hertzian contact and exhibits a strong dependence on normal load, probably due to asperities and oxide films on the steel-ball surfaces. A major conclusion of our study is that scalar transport can serve as a useful macroscopic probe for stress anisotropy and particle-contact topology in granular media.

14. SUBJECT TERMS
Granular media, Quasi-static Mechanics, Transport Properties, Numerical Simulation, Electrical Conductivity and Contact Resistance

15. NUMBER OF PAGES
151 pages

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	Unlimited

Computer Simulation and Experiments on the Quasi-static Mechanics and Transport Properties of Granular Materials

by

Xuejin Zhuang

and

J. D. Goddard (Principal Investigator)

Department of Applied Mechanics

and Engineering Sciences

University of California, San Diego

9500 Gilman Drive

La Jolla, CA 92093-0310

Research Report GR 93-01

to

Air Force Office of Scientific Research

Work performed under

Grant AFOSR F49620-92-J-0037

1 October 1993

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

SUMMARY

The following report, based on the Ph.D. dissertation of the first author [141], presents the results of an improved quasi-static numerical simulation algorithm developed to study both mechanical and scalar transport properties of three-dimensional idealized granular assemblages simultaneously. In addition, the results of an experimental investigation of these properties are also presented and compared against the numerical predictions. The simulation algorithm includes several new techniques, including a shuffling algorithm for the generation of an initial random packing of a granular assemblage and an improved microcell-adjacency method to accelerate particle-contact search. Furthermore, a relaxation method is employed to overcome a singularity in the quasi-linear system of equilibrium equations.

With the objective of correlating scalar transport properties such as electrical conductivity with the mechanical behavior of granular media, we treat the granular assemblage as a resistor network, with particle centers being nodes and interparticle contacts being resistors, for the purpose of computing the conductivity.

The Reynolds dilatancy for randomly dense-packed granular assemblages is found to depend on the interparticle friction, at odds with Reynolds' original hypothesis. The use of linear contact mechanics is found to be valid near the ideal rigid-particle limit. Also, a strong correlation is found between electrical conductivity, stress and fabric tensors, indicating that the scalar transport properties can serve as a useful macroscopic probe for the particle-contact topology in granular media.

Triaxial compression tests, employing steel balls as electrically conductive granular particles serve to confirm our simulation of both the mechanical and scalar transport properties, provided that the electrical conductivity calculations are based on the experimental load-resistance characteristics of individual contacts. The measured contact resistance between steel balls is found to be much higher than theoretical predictions based on Hertzian contact, and exhibits a much stronger dependence on normal load, possibly due to asperities and oxide films on the steel-ball surfaces.

Chapter 1

Introduction

Granular media are materials composed of distinct particles which can move independently of one another and which interact only at highly localized interparticle contact regions. In general, a test on real granular media such as sand is difficult to interpret since the stress inside the sample can not be measured directly and must be estimated from the boundary conditions, although measurements of strain have been made possible by techniques such as X-ray photography technique. Also, Dantu and Wakabayashi (1957) suggested the use of an photoelastic material for rods or discs in order to determine stresses in granular media. Analysis of the force distribution in such a test was first described by De Josselin De Jong and Verruijt (1969), and the technique has been adopted by many researchers [45,78,79,102,103,124]. Although testing of assemblages of photoelastic discs allows for an accurate determination of contact forces, displacements and rotations of the individual discs, the analysis is time consuming. Moreover, the technique is not as yet applicable to 3-dimensional samples.

While physical models are certainly the ultimate test of any physical theory, numerical simulation has the advantages over real experiments in that any microscopic information essential to the understanding of the macroscopic behavior of these systems is accessible at any stage of a test, and "experiments" can be performed numerically that would be very difficult physically. Many reported works show that

numerical techniques are capable of reproducing qualitatively the overall continuum mechanical behavior of real granular materials [39,40,126,128,29]. Compared with real granular media such as soils, however, current numerical techniques are able to simulate only idealized particle shapes such as disks, spheres, ellipsoids etc., in a limited sample size, though the size effect is partly overcome by the use of periodic cell models.

Currently, there are mainly two classes of numerical technique employed to simulate the quasi-static mechanics of granular materials, namely, dynamic and quasi-static. The dynamical simulation technique, often referred to as the Distinct Element Method(DEM) in the older literature, was first developed by Cundall and Strack [39] and has been widely employed since [40,126,41,128,18,33,10]. However, various artificial damping procedures have to be used to suppress parasitic particle vibrations in order to achieve quasi-static conditions. Moreover, it has been noticed recently that the algorithm is only conditionally stable [10,30]. For this reason among others, a direct quasi-static simulation has been receiving increased attention in recent years [115,76,26,14,58].

Reynolds dilatancy, one of the most fundamental characteristics of granular materials, has been accounted for in the mechanical modeling of granular flow [Reynolds(1895), Rowe(1962), Oda(1974a), Nemat-Nasser(1980) and Goddard et al. (1990b)]. The factors influencing dilatancy have been studied by many investigators [120,94,39,103,18,33,14]. The effects of interparticle sliding friction μ and of factors such as initial void ratio and state of packing have been explored sporadically, often with conflicting conclusions. As for randomly dense-packed granular assemblages, Reynolds suggested that friction should have no effect other than to stablize otherwise unstable granular configuration [107]. Skinner showed experimentally that friction has little effect on dilatancy of random assemblages of spherical particles [120]. On the other hand, this effect has also been investigated by means of certain computer simulation, mostly on the random dense-packed granular arrays, which interestingly led to the opposite conclusion. In simulating the physical experiments

reported by Oda and Konishi [98], Cundall et al. found that dilatancy depends on interparticle friction, the assembly with higher friction coefficients dilated more and at a greater rate [39]. Similar results have been reported by Bathurst et al. [18], and Chen [33].

Due to their discrete nature, the behavior of the granular media generally depends on a variety of factors, such as void ratio, interparticle friction, particle shape, and microstructural arrangement or "fabric", to name only a few. Granular fabric is believed to be one of the most important factors determining the overall mechanical response of a medium to the deformation. Oda and Konishi [94,95,96,98] performed direct measurement of fabric in sand specimens and made many important discoveries on the deformation mechanisms of granular materials. However, such measurements are difficult or tedious to perform experimentally. It would, therefore, be highly desirable if the granular fabric could be related to and measured indirectly by means of macroscopic quantities. Dynamic shear modulus and even the complete set of elastic moduli, inferred from wave speed measurements, has been found to contain direct information about the internal fabric [31,32,66,1,2].

On the other hand, scalar transport processes such as electrical or thermal conduction through granular materials can provide another such macroscopic quantity, since the effective conductivity of granular materials depends not only on the conductivities of solid grains and interstitial or pore fluid, but also on the volume fraction of solid particles (void ratio) and particle arrangement or fabric. In fact, the evolution of mechanical anisotropy of water saturated sands and clays has been studied in triaxial compression tests by monitoring the radial and axial electrical conductivity [85,8,4,5]. However, since sand grains are themselves not electrically conductive, the current is conducted only through the pore water. Therefore, the measured anisotropy of conductivity mainly reflects the anisotropy of the void space. While void space is part of the internal structure, it does not serve as a good indicator of granular contact topology. In particular, granular chain structures, which bear the major load, and the variation of interparticle contact forces are not fully captured by

the conductivity measurements of many earlier studies [85,8,4,5].

Based on above considerations, the present work is concerned with a system consisting of electrically conductive particles and an electrically nonconductive interstitial fluid. One objective is to find correlation between the scalar property and mechanical properties during deformation. The investigation includes both numerical simulation and physical experiments. The numerical simulations allow access to detailed microstructural information, such as internal fabric, coordination number, local contact force etc.. The triaxial compression tests, which employ steel balls as conductive granular particles serve to validate the computer simulations.

In Chapter 2, some important aspects of the theoretical development and its application to current investigation will be reviewed briefly, including: (1) the fabric tensor, (2) particle contact mechanics and nonlinear elasticity of granular media, and (3) scalar transport through granular media. Chapter 3 provides a detailed description of the quasi-static simulation and the various newly developed simulation techniques, while Chapter 4 covers the experimental aspects of the current investigation. Numerical simulations, mainly aimed at the study of the microstructural properties of the media, are explored in Chapter 5. Next, the results of computer simulation on scalar transport through idealized granular assemblages are compared with experimental observations in Chapter 6. Finally, Chapter 7 summarizes the major conclusions of the present study and suggestions for future work.

Chapter 2

Literature Review

2.1 Microstructure of Granular Media

It is well accepted nowadays that porosity or solid volume fraction alone is not sufficient to characterize the geometry of the local microstructure of a granular material, given that two specimens of a granular material such as soil, with identical porosity, may possess quite different microstructure and behave mechanically in entirely different ways. In order to understand the dependence of the stress-strain relation on microstructure, additional geometric measures of local structure such as the geometric fabric tensor, have been proposed by many investigators in different fields, including granular materials, soil and rock mechanics [99,100,113,73], cancellous or spongy bone mechanics [62], composite micromechanics [52] etc..

Oda (1978) [99] introduced the concept of a fabric ellipsoid, an ellipsoid determined by the three dimensional distribution of the unit normal to the tangential contact planes. Oda, Konishi and Nemat-Nasser (1980) [100] developed the idea of the fabric ellipsoid, equivalent to a second rank symmetric tensor, and argued that, after porosity, it is the second best measure of microstructure in granular materials, which appears to be a matter of general agreement now. Following the work of Oda et al., these second rank tensors are generally called fabric tensors.

According to Oda (1978), fabric represents the spatial arrangement of parti-

cles and associated voids. This may includes: (a) orientation fabric, which relates to the inclination of a characteristic dimension of individual particles relative to a reference direction; (b) packing or mutual relation of particles, defined by the probability density function, $E(\mathbf{n})$, of contact normals $\mathbf{n}$, and the average coordination number (the number of contacts per particle).

The anisotropy of granular materials, measured by the fabric tensor, has been divided into two categories: (a) inherent anisotropy, a physical characteristic inherent in the virgin materials and entirely independent of the applied strain; (b) induced anisotropy, due exclusively to the strain associated with an applied stress. Experimental evidence [94] has shown that the mechanical behavior of granular media is greatly affected by their anisotropy which is closely related to the spatial arrangement of its particles and the fabric. Knowing the mechanism for change of fabric during deformation will provide better insight into the evolving anisotropy of granular materials. Therefore, the general concept and the several common measures of fabric will be reviewed in the following.

The precise definition of a fabric tensor varies with the type of material and, sometimes for the same material, according to investigator. The choice of a particular fabric measure is a matter of convenience and its suitability is judged by comparison with experimental observation [124]. A relatively universal second-order moment tensor defined by

$$N_{ij} = < n_i n_j > \tag{2.1}$$

where $< ... >$ designates the sample mean, i.e. $< n_i n_j >= \frac{1}{C_N} \sum^c n_i n_j$, is called the *anisotropy tensor* by Satake (1982) or the *fabric tensor of the first kind* by Kanatani (1984). In eq. 2.1, the n_i are direction cosines of the cth contact normal $\mathbf{n} = (n_i)$ to the tangent plane, with respect to the orthogonal coordinate system. C_N is the total number of contacts in a given volume, and $\mathbf{N} = (N_{ij})$ is symmetric with unit trace.

For non-spherical granules, Nemat-Nasser et al. (1983) proposed the tensors:

$$H_{ij} = < m_i m_j > \tag{2.2}$$

or

$$H_{ij} =< m_i n_j > \tag{2.3}$$

where m_i is the Cartesian components of a unit branch vector, a branch being defined as the connection from the centroid of one particle to that of another touching particle. They even suggested the inclusion of average branch length $\hat{l}$ and contact area $\hat{a}$ into the fabric tensor, represented by Eq. 2.2 and 2.3, to reflect additional information on the microstructure. Higher order fabric tensors, such as $< n_i n_j n_k n_l >$, $< m_i m_j m_k m_l >$ and $< n_i n_j m_k m_l >$, may also be considered. The higher order tensors provides more information regarding the details of the anisotropy (Kanatani 1984 and Subhash et al. 1991).

Kanatani (1984) proposed a distribution density function $E(\mathbf{n})$, defined as

$$E(\mathbf{n}) = \eta F_{i,j,\ldots,k} n_i n_j \ldots n_k \tag{2.4}$$

in which η equals to $1/2\pi$ for two dimensional case and $1/4\pi$ for three dimensional case, $F_{i,j,\ldots,k}$, a tensor of even rank r, is referred to as the "rank r tensor of the second kind". Then,

$$E(\mathbf{n}) = E(-\mathbf{n}) \tag{2.5}$$

$$\int_\Omega E(\mathbf{n}) d\Omega = 1 \tag{2.6}$$

where Ω is the surface of unit sphere, and $E(\mathbf{n})d\Omega$ is the relative number of normals $\mathbf{n}$ falling in the solid angle $d\Omega$, about the direction $\mathbf{n}$. To represent the density function $E(\mathbf{n})$ by the second rank fabric tensor of the second kind, F_{ij}, we have, for the three dimensicnal medium,

$$E(\mathbf{n}) = \frac{1}{4\pi} F_{ij} n_i n_j \tag{2.7}$$

$$F_{ij} = \frac{15}{2}(N_{ij} - \frac{1}{5}\delta_{ij}) \tag{2.8}$$

$i, j = 1, 2, 3$, and for a two dimensional medium,

$$E(\mathbf{n}) = \frac{1}{2\pi} F_{ij} n_i n_j \tag{2.9}$$

$$F_{ij} = 4(N_{ij} - \frac{1}{4}\delta_{ij}) \tag{2.10}$$

for $i, j = 1, 2$, in which δ_{ij} denotes Kronecker's delta.

2.2 Contact Mechanics and Nonlinear Elasticity

Cohesionless granular materials support an ambient shear stress only through the contact between particles. Therefore, it is plausible that the mechanism of local contacts should have great influence on overall mechanical properties of these media. One example is the apparent nonlinear elasticity at small strains exhibited collectively by an assemblage of particles which behave individually in a linear elastic way. This effect can be ascribed to the intrinsic nonlinearity of the contact mechanics governing particle–particle interactions (Goddard 1990).

Hertz first initiated the mathematical study of the effects produced by mutual compression of elastic bodies for the case in which the forces between bodies are normal to the contact surfaces [84,72]. Considering two elastic spheres in contact, according to Hertzian theory, a circular contact surface is produced, with radius given by

$$a = (M_1 f_n R)^{\frac{1}{3}} \tag{2.11}$$

where f_n is the normal force, R is the radius of the spheres, and $M_1 = 3(1-\nu^2)/4E$, in which E and ν are Young's modulus and Poisson's ratio of the material, respectively. The theory also gives the relative approach of the spheres

$$\delta = 2(M_1 f_n / R^{1/2})^{2/3} \tag{2.12}$$

Hence the apparent normal contact stiffness is given as

$$k_n = \frac{df_n}{d\delta} = \frac{3}{2}(\frac{R}{M_1^2})^{1/3} f_n^{1/3} \tag{2.13}$$

The tangential stiffness for frictional contacts under oblique contact force was given by Mindlin and Deresiewicz [84,26]:

$$k_t = M_2 k_n (1 - \frac{f_t}{\mu f_n})^{1/3} \tag{2.14}$$

where $M_2 = 2(1-\nu)/(2-\nu)$, μ is the interparticle friction coefficient and f_t is the resultant shear force at the contact.

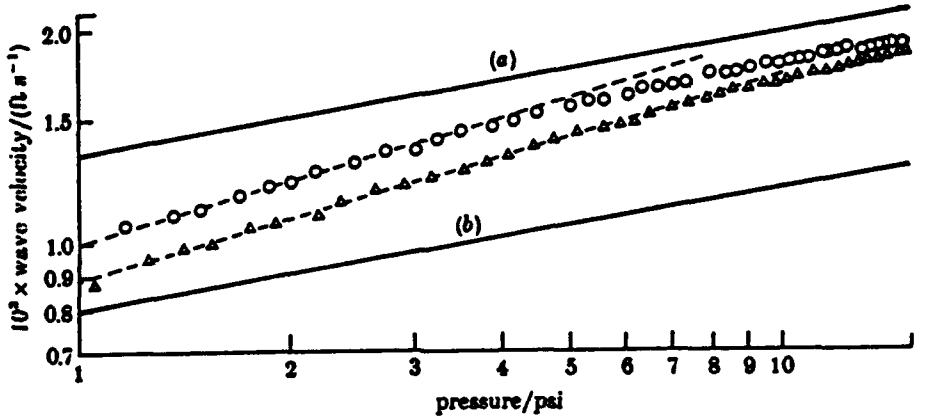


Figure 2.1: Elastic wave velocity in an FCC packing of $\frac{1}{3}$ inch diameter steel balls with 'low' (Δ) and 'high' (o) dimensional tolerances, $\pm 50 \times 10^{-6}$ inches and $\pm 10 \times 10^{-6}$ inches, respectively; after Duffy and Mindlin (1957). The broken lines of slope $\frac{1}{4}$ have been added by Goddard (1990c) for comparison. The solid lines with slope $\frac{1}{6}$ represent the Hertz-Mindlin contact, (a) with, and (b) without tangential stiffness. With permission of the author of [57].

The above theory has been adopted in most theoretical treatments of the micromechanics of granular media, dating from the landmark works of Mindlin and co-workers ([46] etc.) up to the most recent publications on the subject ([44,134] and the references cited). As suggested by above equations, the underlying theory leads inevitably to the power-law scaling $E \sim p^{\frac{1}{3}}$ for the dependence of various elastic moduli on confining stress p and, hence, to the scaling $v \sim p^{\frac{1}{6}}$ for various elastic wave velocities v (with magnitudes characterized by $\sqrt{E/\rho}$, here ρ is the material density). However, experimental evidence within soil mechanics and geophysics shows that the scaling $E \sim p^{\frac{1}{2}}$ and $v \sim p^{\frac{1}{4}}$ are much more representative (see Goddard [57] for a complete survey), although the pressure dependence may change from $p^{1/2}$ to $p^{1/3}$ in high pressure regime or under prolonged vibration at large amplitudes. Such observations are illustrated in figure 2.1.

It is also reflected quantitatively in the widely used empirical formula for the shear modulus of dry sands (see [57] and references cited) under isotropic confinement

at initial or base pressure p:

$$G = \xi[(e_c - e)^2/(1 + e)]p^{1/2} \tag{2.15}$$

where e is void ratio, while ξ and e_c are constants.

In a detailed analysis [57], Goddard showed that one can explain such frequently observed departures from the 1/6-power dependence predicted by Hertz–contact theory on the basis of two rather distinct hypothesis. The first involves nonhertzian asperities while the second appeals to the nonlinearities arising from strain–induced changes in the number of particle contacts. For isotropic confinement both the above hypothesis yield a 1/4-power dependence of wave speed on pressure at low confining pressures, with a transition to a 1/6-power dependence at high pressures.

2.3 Conduction through Granular Materials

2.3.1 Introduction

The prediction of the effective conductivity of two-phase media, in which one phase is dispersed in a second, has occupied engineers and physicists for the past one hundred and twenty years [130,35,16,9,47,22,13,55]. This long interest has been fueled by the proliferation of man-made composite materials and the need to predict bulk properties such as effective conductivity. Maxwell (1873) was the first to theoretically calculate the effective conductivity of a dilute stationary suspension of spherical particles. By considering only the interaction of a single sphere in a potential gradient, Maxwell was able to obtain the following well-known equation

$$\frac{k^*}{k_0} = \frac{1 + 2\beta\phi}{1 - \beta\phi} \tag{2.16}$$

where, $\beta = (\alpha - 1)/(\alpha + 2)$; α is the ratio of conductivity of the solid particle to that of the matrix(or fluid phase); k^* is the effective conductivity of the suspension; k_0 is

the conductivity of the matrix; ϕ is solid volume fraction. To the order of terms in ϕ to which it is exact, Eq. 2.16 takes the form:

$$\frac{k^*}{k_0} = 1 + 3\beta\phi + O(\phi^2) \tag{2.17}$$

A hundred years later, Jeffrey (1973) extended Maxwell's result to $O(\phi^2)$ by the addition of two-sphere interactions for a random hard-sphere dispersion

$$\frac{k^*}{k_0} = 1 + 3\beta\phi + \hat{\beta}\phi^2 + O(\phi^3) \tag{2.18}$$

where $\hat{\beta} = \hat{\beta}(\beta)$ is a slowly-convergent infinite series in β.

Some progress has also been made for densely packed suspensions of perfectly conducting spheres. Keller (1963) solved this problem correct to $\ln\epsilon$, where ϵ is the dimensionless gap width, for a densely packed simple cubic array of spheres. Batchelor and O'Brien (1977) extended Keller's work to include touching spheres and near-perfect conductors by using a mean-field approach. They theoretically treat the thermal or electrical conduction through static particulate media in the limit of maximum volume fraction, for which the particles make point contact with each other and even interface with flat, convex or concave surfaces under external load. They find that when $\alpha >> 1$ the effective conductivity of random two phase media is given by

$$\frac{k^*}{k_0} = 4\ln\alpha - 11 \tag{2.19}$$

with constant 4 predicted by the theory and the additive constant chosen to achieve a reasonable fit with a variety of experimental data points. Their theory suggests that the exact method of forming a dense suspension will strongly affect its effective conductivity by the resultant average coordination number of the particles, and it illustrates that the microstructure has a measurable effect on the conductivity of a suspension.

2.3.2 Electric Contacts

In present study, we will consider a simplified two phase medium with the continuous phase being nonconductive. Electrical conduction through a packed bed

of steel balls, with air filling the interstitial void, represents a prime example. Under compressional loading, the particles are pressed together and, if elastic, will deform slightly and will develop a flat circle of contact. According to the Hertz theory described in section 2.2, two touching elastic particles which are spherical locally with radius R will develop a flat contact circle of radius

$$a = [\frac{3(1-\nu^2)f_n R}{4E}]^{\frac{1}{3}} \tag{2.20}$$

where a compression force f_n acts on each particle normal to the common tangent plane at the point of contact. Since the pore fluid is non-conductive, electric flux is only possible through the contact circle, and the distribution of potential inside the two particles is approximately the same as that of the velocity potential in irrotational flow of an incompressible fluid through a circular hole in a plane wall [16]. The solution to this latter problem is known, and shows that the normal flux density at the contact circle is

$$J = \frac{k_p \Delta\Phi}{\pi(a^2 - r^2)^{\frac{1}{2}}} \qquad (r < a) \tag{2.21}$$

and the total current across the circle of contact is

$$Q = \int_0^a J 2\pi r dr = 2ak_p\Delta\Phi \tag{2.22}$$

where k_p is the conductivity of the particle material and $\Delta\Phi$ is the difference in electrical potential between the particles. From the above equation, the contact resistance between two particles across the the contact circle is

$$R_c = \frac{1}{2ak_p} \tag{2.23}$$

which is nothing but the constriction resistance of the small flat contact area. A similar equation was also obtained by Holm (1967) and Yovanovich (1967).

However, the contact resistance between two real surface is far more complicated. To understand why, it is necessary to consider the nature of solid surfaces and the effect of foreign materials on the overall resistance. Contact surfaces are irregular on a microscopic scale. Even nominally plane surfaces have a waviness with peak-to-valley dimensions typically from tenths to several micrometers [7,60,6]. When two

contacts are brought together under low loads, they touch at only a few asperities (multispot contact). As the load is increased, more asperities come into contact and the surfaces move together. Therefore, the true area of contact depends on normal load and the hardness of the material [64,61]. This area is only a small fraction of the apparent contact, except at very high loads where the surface can be severely deformed. Furthermore, when the metal surface is exposed to the environment, a contaminant film will be developed through processes such as oxidation and corrosion, particulate contamination (airborne and wear debris), fretting, etc., and soon covers up the virgin metal [136,6]. This contaminant film is often extremely nonconductive, therefore, preventing electrical conduction through the contact. Under such circumstances, conduction is not possible if the film is unbroken, except when the film is only a few atomic layers thick, such that some electron current can penetrate it by means of the *tunnel effect* [64]. Under a normal mechanical load, the insulating film on the contact asperities deforms plastically and fractures, so that pure metal substrates are once again exposed to each other. Figure 2.2 schematically illustrates this situation where the apparent area of contact, the metallic regions, and the places with insulating layers are differentiated. The lines of current flow converge at the region of metallic contact, called "a" spot, as illustrated schematically in Figure 2.3. Contact resistance decreases with increasing load. The softer and more conductive the metal, the lower the contact resistance will be at a given force.

2.3.3 Multispot Theory of Contact

Generally, a multispot problem is simplified by assuming all of the a-spots to be circular and to lie at distances from each other which are large compared to the radii, thus permitting the assumption of no interference between different a-spots. Thus, the total resistance becomes

$$R_c = \frac{1}{2k_p \sum a_i} \tag{2.24}$$

where subscript 'i' represents the ith a-spot.

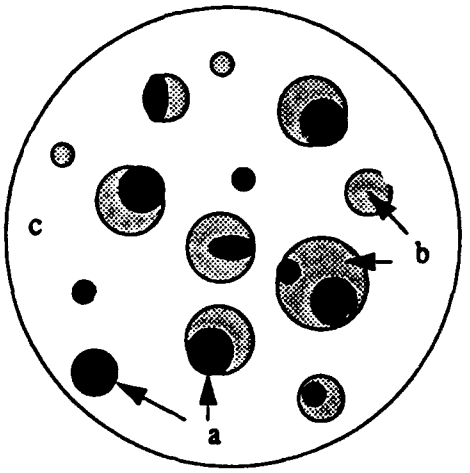


Figure 2.2: Schematic illustration of apparent contact surface. The metallic contact regions *a* are indicated by dark areas. Contact at region *b* (shaded areas) is with insulating contaminant film. Region c does not touch.

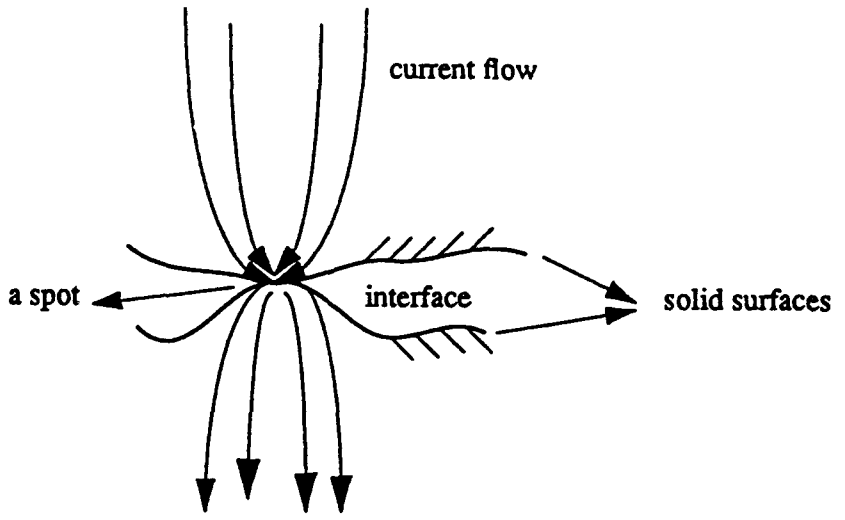


Figure 2.3: Microscopic view of a real contact interface: "a" spot and lines of current flow.

For the case in which the a-spots lie close to each other so that the constricted lines of flow from different a-spots deflect each other, then Eq. 2.24 is no longer valid. Holm [64] has made some approximations for the case of the uniformly distributed a-spots, giving the following expression

$$R_c = \frac{1}{2\pi n a k_p} arctan \frac{\sqrt{l^2 - a^2}}{a} - 0.6 \frac{\sqrt{l^2 - a^2}}{k_p A_r} + \frac{1}{4 k_p r} \tag{2.25}$$

where n is number of a-spots, a the radius of a-spot, $2l$ the average distance between neighboring a-spots, A_r the area of apparent contact, and r the radius of the apparent contact surface.

2.3.4 Effective Conductivity

Suppose that a uniform intensity gradient is set up in the medium, perhaps by imposing uniform and different values of the intensity at two distant parallel boundaries. Although we restrict ourselves to the electrical conduction problem, the formulation can be applied to the transport of other scalar properties such as thermal conduction and mass diffusion. Henceforth we shall use terms and notation appropriate to the case of electrical conduction for convenience. So the mean intensity gradient will be written as $< \nabla\Phi >$, where $\nabla\Phi$ is the electrical potential gradient at a point in the medium (not necessarily lying in the matrix) and the brackets $< ... >$ denote an average over the entire volume of the medium. The local current density $\mathbf{J}$ is equal to $-k_0 \nabla\Phi$ at a point in the matrix and $-k_p \nabla\Phi$ at a point in a particle. At each point on the surface of a particle Φ and the normal component of $\mathbf{J}$ are continuous; and at each point not on such a surface

$$\nabla \cdot \mathbf{J} = 0 \quad and \quad \nabla^2 \Phi = 0. \tag{2.26}$$

Because of the intrinsic linearity, the magnitude of all potential differences are proportional to the magnitude of $< \nabla\Phi >$, and so for the mean flux density we have the linear relation [16]

$$< \mathbf{J} > = -\mathbf{K}^* < \nabla\Phi >, \tag{2.27}$$

where the effective conductivity $\mathbf{K}^*$ is a second-rank tensor, dependent on the structure of the medium.

Next, we will derive a formulation for the mean flux density through granular media with a nonconductive fluid phase. In this case, if particles are highly conductive, the resistivity will be brought about at contact points only. Therefore, the medium can be approximated by a resistor network, as shown in figure 2.4, with particle centers being nodes and interparticle contact being resistor. Similarly, Fig. 2.4 can also be used to represent a elastic network [57] by replacing resistors by elastic springs whenever elastic properties are involved. By definition, we have

$$< \mathbf{J} >= \frac{1}{V} \int_V \mathbf{J} dV \tag{2.28}$$

or

$$< \mathbf{J} >= \frac{1}{V} \int_{V_m} \mathbf{J} dV + \frac{1}{V} \sum^{N} \int_{V_p} \mathbf{J} dV \tag{2.29}$$

where V, V_p and V_m represent the total volumes of medium, particle and fluid phase, respectively, N the number of particles.

For the nonconductive matrix, the first term on the right in Eq. 2.29 vanishes, so we have

$$< \mathbf{J} >= \frac{1}{V} \sum^{N} \int_{V_p} \mathbf{J} dV \tag{2.30}$$

Applying Gauss' divergence theorem to Eq. 2.30, the following is obtained

$$< \mathbf{J} >= \frac{1}{V} \sum^{N} \int_{S_p} \mathbf{x} \mathbf{J} \cdot \mathbf{n} dS \tag{2.31}$$

where S_p is the surface of a particle, $\mathbf{x}$ is the spatial position of the points on S_p, and $\mathbf{n}$ represents the unit normal to S_p. Noting that the term $\mathbf{J} \cdot \mathbf{n} dS$ is current passing through the portion dS of particle surface S_p and assuming that the current passes through the individual particle at discrete points of contact, we obtain

$$< \mathbf{J} >= \frac{1}{V} \sum^{N} \sum^{c} \mathbf{x} Q \tag{2.32}$$

where Q is current flowing through the points of contact. For spherical particles, $\mathbf{x} = \mathbf{x}^p + R\mathbf{n}$, here $\mathbf{x}^p$ is the position vector of the particle centroid, and Eq. 2.32

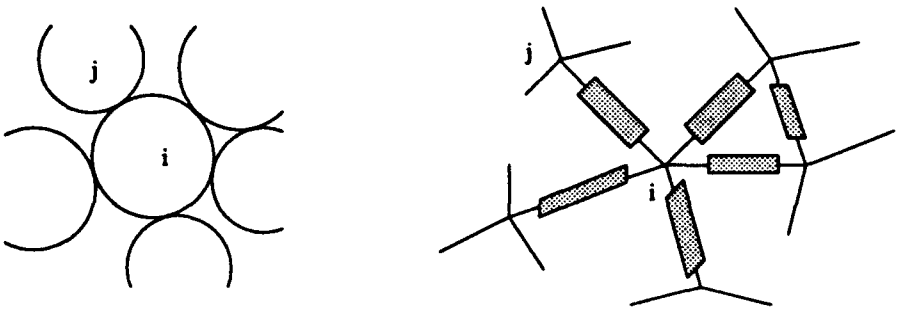


Figure 2.4: Network

becomes

$$< \mathbf{J} >= \frac{1}{V} \sum^{N} R \sum^{c} \mathbf{n} Q \tag{2.33}$$

where, electrical conservation (Kirchhoff's law) gives:

$$\sum^{c} \mathbf{x}^{p} Q = \mathbf{x}^{p} \sum^{c} Q = 0 \tag{2.34}$$

The formulation described above is similar to one to derive the macroscopic stress tensor for a assemblage of granular materials except for the tensorial orders involved.

In order to compute the interparticle current Q, the local potentials Φ of the particles must be determined first. These local potentials can be further divided into two parts: one derived from the mean potential gradient, the other being fluctuation necessary to satisfy the current balance condition within the system. The latter is obtained by solving the system of linear equations

$$\mathbf{A}\Phi' = \mathbf{B} \tag{2.35}$$

where $\mathbf{A}$ represents the conductance matrix, Φ' is the fluctuation and $\mathbf{B}$ is the net unbalanced current owing to the mean potential gradient. When a cluster of one or more particles is isolated from the rest, the matrix $\mathbf{A}$ become singular. This singularity is resolved by means of the relaxation method to be discussed in Section 3.1.3.

From Eq. 2.27 and Eq. 2.34, one can infer the effective conductivity tensor $\mathbf{K}^*$, which represents exact solution of the problem, in contrast to the mean-field theory of Batchelor & O'Brien [16] described next.

Their mean-field theory assumes that the potential difference $\Phi_j - \Phi_i$ between particles i and j is given by $(\mathbf{x}^j - \mathbf{x}^i) \cdot < \nabla\Phi >= -2R\mathbf{n} \cdot < \nabla\Phi >$, i.e. to the difference between the potential at the two sphere centers in the mean potential field. Furthermore, we assume that the contact resistances at all contacts take the identical average value $\bar{R}_c$. With these assumptions, one no longer has local electrical conservation, but rather a global conservation in some average sense.

According to above assumptions, we can write

$$Q = \frac{1}{\bar{R}_c}\Delta\Phi = -\frac{2R}{\bar{R}_c}\mathbf{n} \cdot < \Delta\Phi > \tag{2.36}$$

Combining Eq. 2.36 with Eq. 2.33 leads to the result similar to one given by Batchelor & O'Brien

$$< \mathbf{J} >= -\frac{2R^2}{\bar{R}_c V}\sum^{N}\sum^{c}\mathbf{nn} \cdot < \nabla\Phi > \tag{2.37}$$

Therefore, by comparing Eq. 2.37 with 2.27, one obtains the effective conductivity tensor according to the mean-field theory

$$\bar{\mathbf{K}}^* = \frac{2R^2}{\bar{R}_c V}\sum^{N}\sum^{c}\mathbf{nn} \tag{2.38}$$

or in terms of the fabric tensor $\mathbf{N}$,

$$\bar{\mathbf{K}}^* = \frac{2R^2 C_N}{\bar{R}_c V}\mathbf{N} \tag{2.39}$$

where C_N is the total number of contacts in a given volume, which is equal to N times the average coordination number.

The estimates of effective conductivity provided by the mean-field theory of Batchelor & O'Brien, as outlined in [16], is different from that provided by the conventional Voigt-Reuss-Hill bounds which do not depend on the fabric tensor.

Chapter 3

Quasi-static Simulation and Assemblage Generation

As a continuation and extension to three dimensions of the work of Bashir and Goddard [14], we have developed an improved version of their 2D programs by introducing several new techniques [58]. Among these is the combination of the particle-assemblage generation and the computation of particle motion together into one single program. The new program includes a shuffling algorithm, for generating an initially random loose-packed configuration of particles, and an improved microcell-adjacency method to further accelerate particle-contact search. Furthermore, we have also overcome a singularity in the quasi-linear system of equilibrium equations by means of a relaxation method[121]. Our program is able to simulate any deformation history and allows us to study both mechanical and scalar transport properties of an idealized granular assemblage simultaneously.

The validity of the numerical algorithm is "tested" by comparison against the triaxial compression experiments. I know of no other way to test it except against other numerical codes, however, which have their own problems. The triaxial compression experiments are to be described in Chapter 4, while the qualitative comparisons between the results of numerical simulations and triaxial compression tests will be given in Section 6.2.

The following sections will describe the quasi-static simulation and aforementioned techniques.

3.1 The Model and The Relaxation Method

Unlike dynamic simulations, in which the full (Newtonian) dynamical equations are employed to update particle configurations, the present quasi-static scheme first moves every particle in the system according to the mean deformation gradient, thus destroying the state of equilibrium. Hence, particles have to be relocated to a equilibrium position by means of fluctuations about the mean. These fluctuating displacements of an individual particle are determined by the total unbalanced elastic force exerted on it as a result of the mean deformation. Equilibrium is achieved by an algorithm that allows the system to expand or contract volumetrically when necessary to maintain a control pressure or stress at a desired level which, thereby, allows us to compute the granular dilatancy.

3.1.1 The Force-displacement Law

During the deformation of granular assemblages, particles move with independent degrees of freedom and interact with each other only at their contact points. The assumed force–displacement relationship will be presented here for the case of two spherical particles A and B in contact, as shown in figure 3.1.

Particle radius is denoted by R and its centroid by $\mathbf{X}$. Upon the deformation, a particle undergoes translational and rotational displacement increments $\mathbf{u}$ and $\boldsymbol{\omega}$, respectively. The superscript in Fig. 3.1 and in the sequel denotes a given particle. The unit contact normal vector to the tangential plane, viewed from A to B, is expressed as $\mathbf{n} = (\mathbf{X}^B - \mathbf{X}^A)/|\mathbf{X}^B - \mathbf{X}^A|$. The interaction between the particles depends on the relative motion of the contact points. The vectorial components of relative displacement in normal and tangential direction are written as:

$$\Delta \mathbf{u}_n = (\mathbf{u}^B - \mathbf{u}^A) \cdot \mathbf{nn} \tag{3.1}$$

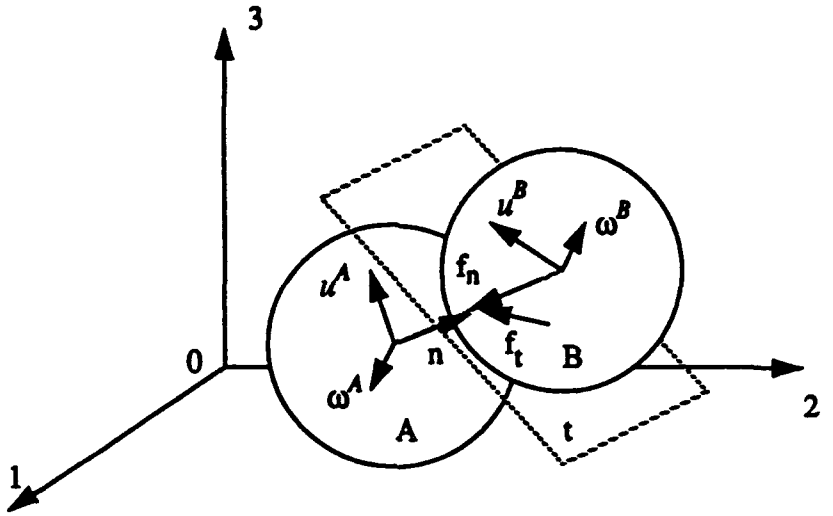


Figure 3.1: Interaction between two particles

and

$$\Delta \mathbf{u}_t = (\mathbf{u}^B - \mathbf{u}^A) - \Delta \mathbf{u}_n + \mathbf{R}^B \times \omega^B + \mathbf{R}^A \times \omega^A \tag{3.2}$$

where $\mathbf{u}^{AB} = \mathbf{u}^B - \mathbf{u}^A$, $\mathbf{R}^A = R^A\mathbf{n}$ and $\mathbf{R}^B = R^B\mathbf{n}$.

These relative displacements are used to calculate increments of normal and shear forces, $\Delta\mathbf{f_n}$ and $\Delta\mathbf{f_t}$, according to:

$$\Delta \mathbf{f}_n = k_n \Delta \mathbf{u}_n \tag{3.3}$$

and

$$\Delta \mathbf{f}_t = k_t \Delta \mathbf{u}_t \tag{3.4}$$

where k_n and k_t denote the normal and tangential elastic stiffnesses, respectively, which may be allowed to depend on $\Delta\mathbf{f_n}$ and $\Delta\mathbf{f_t}$. However, since we are primarily interested in nearly rigid particles, the exact dependence of the elastic stiffnesses on $\Delta\mathbf{f}$ is presumably not important (vide infra).

Furthermore, the force increments $\Delta\mathbf{f}_n$ and $\Delta\mathbf{f}_t$ are added, respectively, to the forces $\mathbf{f}_n^0$ and $\mathbf{f}_t^0$ that existed previously between two particles to yield the current

values:

$$\mathbf{f}_n = \mathbf{f}_n^0 + \Delta\mathbf{f}_n \tag{3.5}$$

and

$$\mathbf{f}_t = \mathbf{f}_t^0 + \Delta\mathbf{f}_t \tag{3.6}$$

The components, $\mathbf{f}_n = f_n\mathbf{n}$ and $\mathbf{f}_t = f_t\mathbf{t}$, of the force vector act along the directions of contact normal and tangent plane and are both set to be zero if f_n is not compressional (since cohesionless particles cannot sustain a tensile force). A (Coulomb) sliding friction law is incorporated as follows: the magnitude of the shear force $\mathbf{f}_t$ given by eq. 3.6 is checked against the maximum possible shear force magnitude:

$$(f_t)_{max} = \mu\,|\mathbf{f}_n| + C_h \tag{3.7}$$

where $\mu(\equiv \tan\phi_\mu)$ is the coefficient of sliding friction (defining the so-called angle of intergranular friction ϕ_μ), and C_h represents cohesion, which is taken as identically zero for the non–cohesive particles considered here. If $|\mathbf{f}_t|$ exceeds $(f_t)_{max}$, sliding occurs at the contact point. Under this circumstance, $\mathbf{f}_t$ takes the value of $(f_t)_{max}$, and maintains its direction. Therefore, the total force and couple exerted on particle A by particle B are given by:

$$\mathbf{f} = \mathbf{f}_n + \mathbf{f}_t \tag{3.8}$$

$$\mathbf{M} = \mathbf{R}^A \times \mathbf{f}_t \tag{3.9}$$

3.1.2 The Governing Equations

The force $\mathbf{f}$ and couple $\mathbf{M}$ are next decomposed into three Cartesian components, which yields in matrix form:

$$\mathbf{F} = \mathbf{F}^0 + \mathbf{k}_{AB}\Delta\mathbf{u}(AB) \tag{3.10}$$

where, $\mathbf{F} = [f_x, f_y, f_z, m_x, m_y, m_z]$, the generalized force, represents the components of force and moment exerted currently on particle A by B. $\mathbf{F}^0 = [f_x^0, f_y^0, f_z^0, m_x^0, m_y^0, m_z^0]$ represents the components of the force and the moment in the previous state. The

matrix $\mathbf{k}_{AB}$ is called the local contact-stiffness matrix, while $\Delta\mathbf{u}(AB)$ is the generalized relative displacement between A and B, written as:

$$\Delta\mathbf{u}(AB) \equiv \begin{pmatrix} u_x^B - u_x^A \\ u_y^B - u_y^A \\ u_z^B - u_z^A \\ r^B\omega_x^B + r^A\omega_x^A \\ r^B\omega_y^B + r^A\omega_y^A \\ r^B\omega_z^B + r^A\omega_z^A \end{pmatrix}. \tag{3.11}$$

where the subscripts denote the corresponding Cartesian component.

For all contacts on A to be in static equilibrium, the sum of generalized force must vanish:

$$\sum_B \mathbf{F} = \sum_B \mathbf{F}^0 + \sum_B \mathbf{k}_{AB}\Delta\mathbf{u}(AB) = \mathbf{0} \tag{3.12}$$

or

$$\sum_B \mathbf{k}_{AB}\Delta\mathbf{u}(AB) = -\sum_B \mathbf{F}^0 \tag{3.13}$$

In the current simulation, the displacement of each particle is additively decomposed into two components: the macroscopically imposed mean $\bar{\mathbf{u}}$ defined by the global velocity gradient and a fluctuation $\mathbf{u}'$, the latter being such that the force balance eq. 3.13 is satisfied. Therefore, eq. 3.13 becomes:

$$\sum_B \mathbf{k}_{AB}\Delta\mathbf{u}'(AB) = -\sum_B \mathbf{F}^0 - \sum_B \mathbf{k}_{AB}\Delta\bar{\mathbf{u}}(AB) \tag{3.14}$$

for $A = 1, 2, \cdots, N$, with N denoting the total number of particles within the system. This represents a system of quasi–linear equations for the $(6N)$ fluctuating particle displacements:

$$\mathbf{K}\mathbf{x} = \mathbf{b} \tag{3.15}$$

where $\mathbf{K}$ is the grand stiffness matrix, $\mathbf{x} = [u_x'(1), u_y'(1), u_z'(1), \omega_x'(1), \omega_y'(1), \omega_z'(1), \cdots, u_x'(N), u_y'(N), u_z'(N), \omega_x'(N), \omega_y'(N), \omega_z'(N)]$ the vector of the fluctuating displacements and rotations, and $\mathbf{b}$ the unbalanced force arising from the mean displacements and forces from the prior deformation step.

3.1.3 Relaxation Method

The **K** matrix in (3.15) becomes singular whenever a cluster of particles is isolated from the remainder, giving rise to "neutral" or "zero-frequency" elastic vibrational modes representing a finite dimensional null space of **K**. To overcome this singularity, Bashir and Goddard add a small artificial term to the diagonal of **K** and then employ a gaussian elimination method to solve (3.15) for **x**.

In the present simulation, we utilize the relaxation method (originally due to Southwell [121]) as our linear-equation solver, since it effectively cuts out the zero-frequency modes of **K**. Being an iterative method, relaxation involves two procedures to accelerate convergence. First, the relaxation order is determined by searching for the residual of greatest magnitude $|R_i|_{max}$ (the residuals R being the difference between the right-hand and left-hand sides of (3.15), evaluated at the current values of **x** in an iteration), then "relaxing" the corresponding equation by calculating a new value of x_i so that $(R_i)_{max} = 0$. This modifies all other residuals, which also depend on x_i. The procedure is applied repetitively until all the residuals satisfy a preset convergence criterion on some norm $|R|$. In the present context the fluctuations determined by the relaxation method serve to move only those particles, or particle clusters, having non-equilibrated forces or moments. Hence, isolated clusters do not fluctuate, and we avoid the singularity in inverting (3.15).

For the packing algorithm described in the sequel, the relaxation method is particularly effective, since in the early stages, the number of particle contacts is small and only those particles not in equilibrium need be moved. Furthermore, the relaxation scheme always finds the maximum unbalanced forces and adjusts particle positions so as to balance them out.

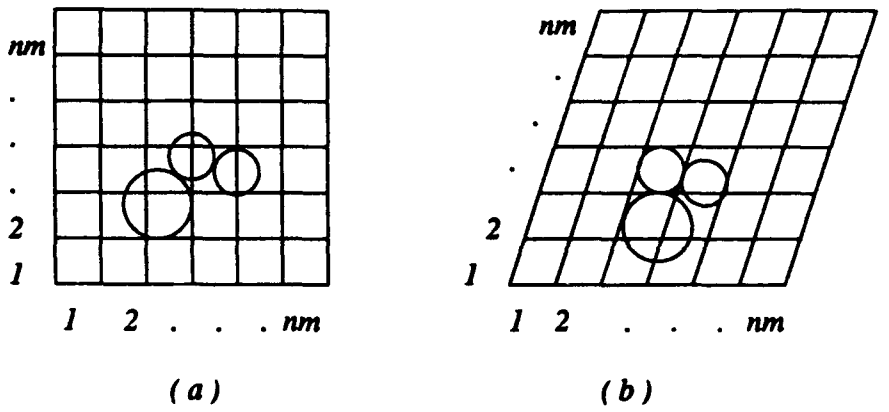


Figure 3.2: Microcells and the simulation cell in (a) initial, and (b) sheared configurations

3.2 The Microcell Method and The Adjacency Matrix

In the computer simulation of a classical mechanical system of N interacting particles, it is generally necessary to search for all particles within the range of spatial interaction of a given particle. In general, one needs $N(N-1)/2$ such searches, including a time-consuming evaluation of particle separations, a non-trivial task when the number of particles is large. However, the search time can be reduced to $O(N)$ by means of spatial microcell methods [3] and the associated adjacency-matrix technique.

In the 2D case, for instance, the deformable simulation cell is divided into regular lattice of $nm \times m$ initially square microcells as shown in Fig. 3.2. A microcell is small enough to contain the center of at most one particle throughout the subsequent cellular deformations. All microcells are then labeled ordinally. For each microcell, the definition of adjacent microcells may include a neighborhood extending several microcell layers outward, depending upon the range of the pair interaction considered. Whatever the range, a matrix Ac defines the adjacency of microcells:

$$Ac(i,j) = \begin{cases} 1, & \text{if microcells } i \text{ and } j \text{ are adjacent} \\ 0, & \text{otherwise} \end{cases} \tag{3.16}$$

where, $i, j = 1, 2, ..., mc(mc = nm \times nm)$, which is nothing more than the connectivity matrix of the associated graph [131]. We next define a second matrix Oc to represent the occupancy of microcells by particles, such that:

$$Oc(i,j) = \begin{cases} 1, & \text{if microcell } i \text{ is occupied by particle } j \\ 0, & \text{otherwise} \end{cases} \tag{3.17}$$

where, $i = 1, 2, ..., mc$, and $j = 1, 2, ..., N$. A third matrix Ap is then used to represent the adjacency of two particles:

$$Ap(i,j) = \begin{cases} 1, & \text{if particles } i \text{ and } j \text{ are adjacent} \\ 0, & \text{otherwise} \end{cases} \tag{3.18}$$

where, $i, j = 1, 2, ..., N$, as determined by their occupancy of adjacent or non-adjacent microcells. This matrix gives the "Verlet neighbor list" of molecular dynamics [3] and can be expressed as the matrix product:

$$Ap = Oc^T AcOc \tag{3.19}$$

Once the microcell adjacency matrix Ac is established, it remains unchanged as long as the microcell topology is fixed during the simulation. Upon determination of the occupancy matrix Oc at each deformation step, the particle adjacency matrix Ap can be found easily by the simple operation (3.19). However, (3.19) is computationally equivalent to:

$$Ap(i,j) = Ac(map(i), map(j)) \tag{3.20}$$

where map defines a mapping array whose element $map(i)$ equals the ordinal number $(1, 2, ..., mc)$ of the microcell occupied by particle i and which, therefore, corresponds to the row vectors of Oc. Based on the computed particle adjacency matrix Ap, the

current program searches for all particle contacts in order to construct the stiffness matrix and to calculate contact forces.

In our simulation on rigid spheres, the size of the cubical microcell is chosen such that its largest diagonal is equal to the smallest particle diameter in the system to assure that not more than one particle simultaneously occupies a given microcell. Furthermore, the largest particle diameter defines a cutoff distance at which one must search for a potential contact with a neighboring particle. For a 2D monodisperse disk assemblage, therefore, two surrounding layers will be sufficient to cover the cutoff distance, which means there are 24 microcells adjacent to each microcell. At the start of a simulation, the microcell adjacency matrix Ac is constructed accordingly and remains unaltered throughout the computation. At each deformation step, whenever particles move to new positions the mapping array is updated, which is a rapid process. For a given particle, we need only look at the neighboring 24 microcells surrounding its microcell to find the adjacent particles. In the worst case, 24 searches would be required if all neighboring microcells were occupied. Therefore, $12N$ provides an upper-bound on the total searches necessary if we consider a pair of neighboring particles as one search.

In reality, the number of searches required depends upon the number of particles lying within the cutoff distance or upon the system density and configuration. In a 2D random particle assemblage, the average number of necessary searches is far less than 24 per particle. Through our computations, we have found that the average number of searches for each particle is about 6 for random loose-packed, and 11 for random dense-packed monodisperse disk systems. Therefore, the total number of searches is approximately $3N$ and $5.5N$, respectively.

3.3 Random Configuration Generation

In the past thirty years or so, the packing of disks and spheres of equal radii in 2D and 3D has been studied extensively by both experimental and theoret-

ical means, in part because they serve as useful models for a variety of amorphous materials such as molecular fluids and glasses. The macroscopic properties of granular materials and porous media have also been modeled by sphere packings. Three models which are commonly employed for packings of disks and spheres are the dense ordered packing, dense random packing, and loose random packing. The dense ordered packing for rigid spheres of equal radii occurs when the density is equal to 0.7405 in 3D(FCC or HCP). Similarly, the density is equal to 0.9069 in 2D(triangular). For dense random packings, it is generally believed that the densities fall into a range 0.62 to 0.66 for 3D and 0.81 to 0.87 for 2D [19,50,71,114,133].

In a previous study of 2D disk assemblages, Bashir and Goddard [14] have employed two distinct algorithms to generate two types of assemblages: imperfect triangular close-packed for the monodisperse assemblage and pseudogravitational for the polydisperse. Recognizing the limitation of those algorithms in allowing for variable initial densities and for generating random isotropic configurations, we have developed a packing algorithm which is capable of densifying an initially random loose configuration to any desired density for both monodisperse and polydisperse assemblages by means of cyclic shear under isotropic confining pressure. One could if desired add body forces such as gravity but we shall not consider them here.

3.3.1 The Shuffling Algorithm

There are many ways of realizing random sequences, the conventional one being the standard random-number generation. For the purpose of generating random particle assemblages, we introduce a new way of rapidly obtaining repeated random sequences of numbers by means of a card-shuffling algorithm. The idea is to degrade the order of a given set of numbers by means of a certain number of "riffle" shuffles.

In shuffling theory [43], a single riffle shuffle can generate at most two increasing sequences for an ordered n-member set S, where a increasing sequence is defined to be a sequence whose members are in the increasing order of their ordinal numbers in the original set S. If $F_n(R)$ be the number of permutations of n items

with exactly R increasing sequences, it can be shown that [108]:

$$F_n(R) = \sum_{j=0}^{R} (-1)^j \binom{n+1}{j} (R-j)^n \tag{3.21}$$

Shannon's theorem [51] states that a permutation with exactly 2^k increasing sequences can be obtained by k riffle shuffles of the original set members in only one way.

Hence, in k riffle shuffles, the total number of permutations that can be achieved is:

$$T_n(k) = \sum_{R=1}^{2^k} F_n(R) \tag{3.22}$$

and thus, the number of riffle shuffles k, sufficient to generate a sequence of random numbers is given by [43]:

$$T_n(k) > n! \tag{3.23}$$

(3.23) implies that for a deck of $n = 52$ cards, $k = 7$ riffle shuffles are sufficient to obtain a nearly random sequence [43].

In order to compare our shuffling algorithm with a standard random-number generator, we have calculated the autocorrelation between the shuffled sequence of fifty numbers and an initial ordered sequence, as well as the autocorrelation between two random sequences obtained by a random number generator. Let $S(i)$ represent the elements, $i = 1, 2, ..., n$, of an n-member sequence, then, we employ as autocorrelation function between two such sequences $S^{(1)}$ and $S^{(2)}$ the formula:

$$A = \frac{\sum_{i=1}^{n} sin[\frac{2\pi}{n}(S^{(2)}(i) - \frac{1}{2})] sin[\frac{2\pi}{n}(S^{(1)}(i) - \frac{1}{2})]}{\sum_{i=1}^{n} sin^2[\frac{2\pi}{n}(S^{(1)}(i) - \frac{1}{2})]} \tag{3.24}$$

which treats the sequences as cyclical. We have also measured the CPU time required by both techniques, and the details will be discussed below.

3.3.2 Shuffling vs The Random Number Generator

In our shuffling algorithm, a variant of the riffle shuffle is used, wherein each shuffle consists of one random "cut" and "flip", and one interlacing shuffle [43]. To compare our shuffling algorithm with a random number generator, we have computed

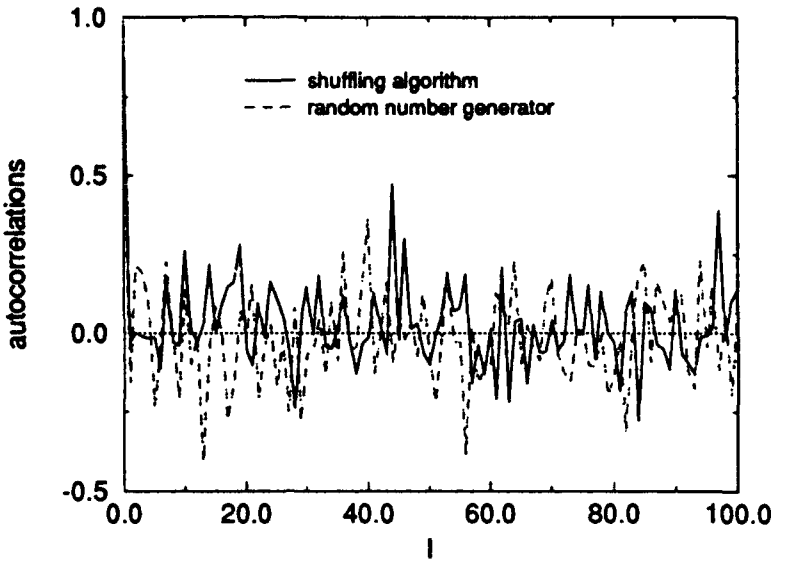


Figure 3.3: A comparison between the shuffling algorithm and random number generator

the autocorrelations between the shuffled sequences of fifty numbers and the initial ordinal sequence as a function of number of shufflings. Similarly we also computed the autocorrelations between the first, second and succeeding sequences with the first sequence of fifty random numbers generated by a random number generator in our computer(a HP-730^{TM} workstation). Fig. 3.3 indicates that sequences of random number generated by the shuffling algorithm are as good as those obtained by the random number generator in terms of randomness, even when we employ only seven riffle shuffles for all of our computations. To compare performance in speed, we have measured the CPU time required by both techniques. Fig. 3.4 clearly shows that the random number generator uses approximately four times as much CPU time as the riffle shuffle.

3.3.3 Initial Random Loose Configurations

In our packing algorithm the size of microcell is chosen sufficiently small so as to contain the center of not more than one single particle under any circumstance,

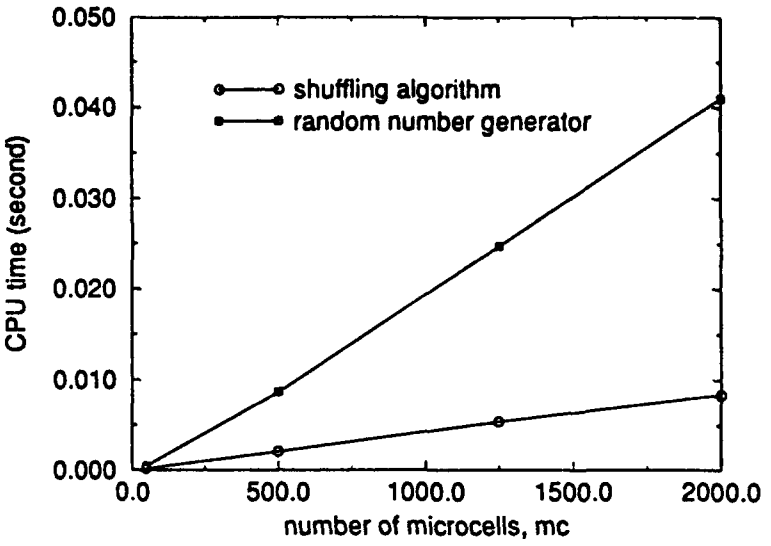


Figure 3.4: A comparison of speeds between the shuffling algorithm and random number generator

but also sufficiently large so as to minimize the total number of microcells.

Fig. 3.5 shows a 2D rectangular microcell $ABCDA$ with two sides being denoted by Δx and Δy. When subject to a simple shear, the microcell $ABCDA$ is deformed to $ABC'D'A$. If the largest dimension AC' is chosen to be equal to the smallest particle diameter we are assured that no two particles can simultaneously occupy the same microcell throughout the deformation. Therefore, we have:

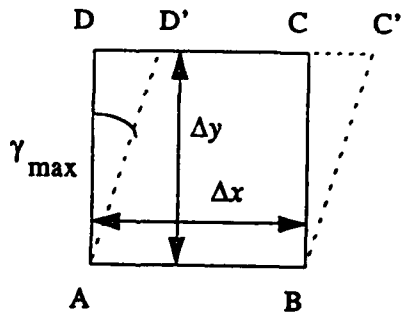


Figure 3.5: Microcell geometry

$$\Delta y = \frac{d_{min}}{\sqrt{(\gamma_{max} + r_{xy})^2 + 1}} \tag{3.25}$$

$$\Delta x = \Delta y \cdot r_{xy} \tag{3.26}$$

where, d_{min} is the smallest particle diameter, γ_{max} is the magnitude of the total shear strain, and r_{xy} is the ratio of two sides ($r_{xy} = 1$, usually).

By assuming an initial density and knowing the total particle volume, we estimate the size of the simulation cell and divide the cell into a lattice of $nm \times nm$ microcells. The microcells are then labeled ordinally from 1 to mc ($=nm \times nm$). To place N particles randomly in the simulation cell, we generate a random sequence of microcells by employing the shuffling algorithm described before. We then pick a microcell from the random sequence and place a particle randomly within the microcell, so as to avoid overlap with previously placed particles, until all N particles are placed successfully. Otherwise, the number of microcells is increased and whole process is repeated. Fig. 3.6 and 3.7 show one such random loose-packed configuration for 132 disks and its corresponding random dense-packed configuration. Similar configurations for 48 polydisperse(multiple-sized) spheres are displayed in Fig. 3.8 and 3.9.

3.3.4 Radial Distribution Functions

The radial distribution functions $g(r)$ for the monodisperse assemblages, r being scaled on particle diameter, have been computed and compared with those generated by the Percus-Yevick (P-Y) equation of statistical mechanics and by a Monte Carlo (M-C) simulation [127,132,137], to verify that both our loose-packed configuration and dense-packed systems are random for 2D as well as 3D assemblages. Fig. 3.10 shows the smoothed $g(r)$ distribution function for 100 realizations of an initially loose-packed configuration of 132 disks(density=0.43). Fig. 3.11 shows the smoothed $g(r)$ for 100 realizations of dense-packed configurations of 132 disks(density=0.80, close to

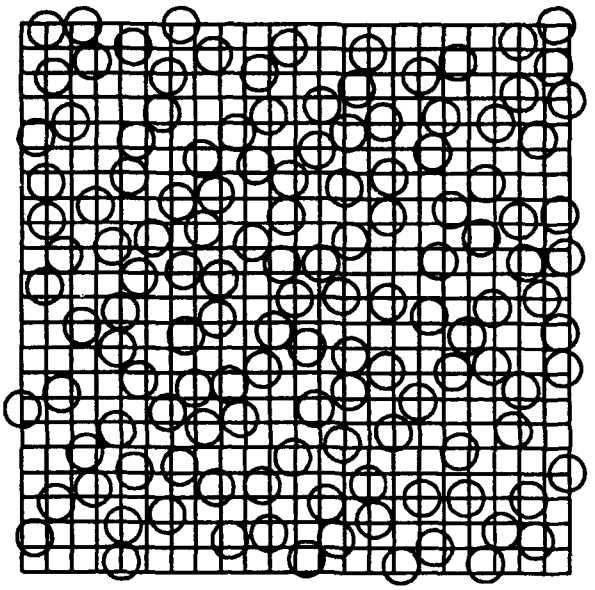


Figure 3.6: Random loose-packed configuration for 132 disks(density=0.43)

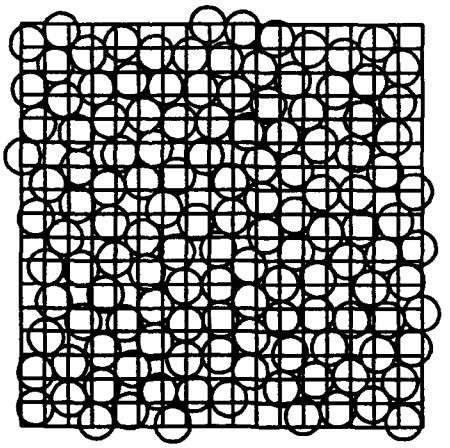


Figure 3.7: Random dense-packed configuration for 132 disks(density=0.80)

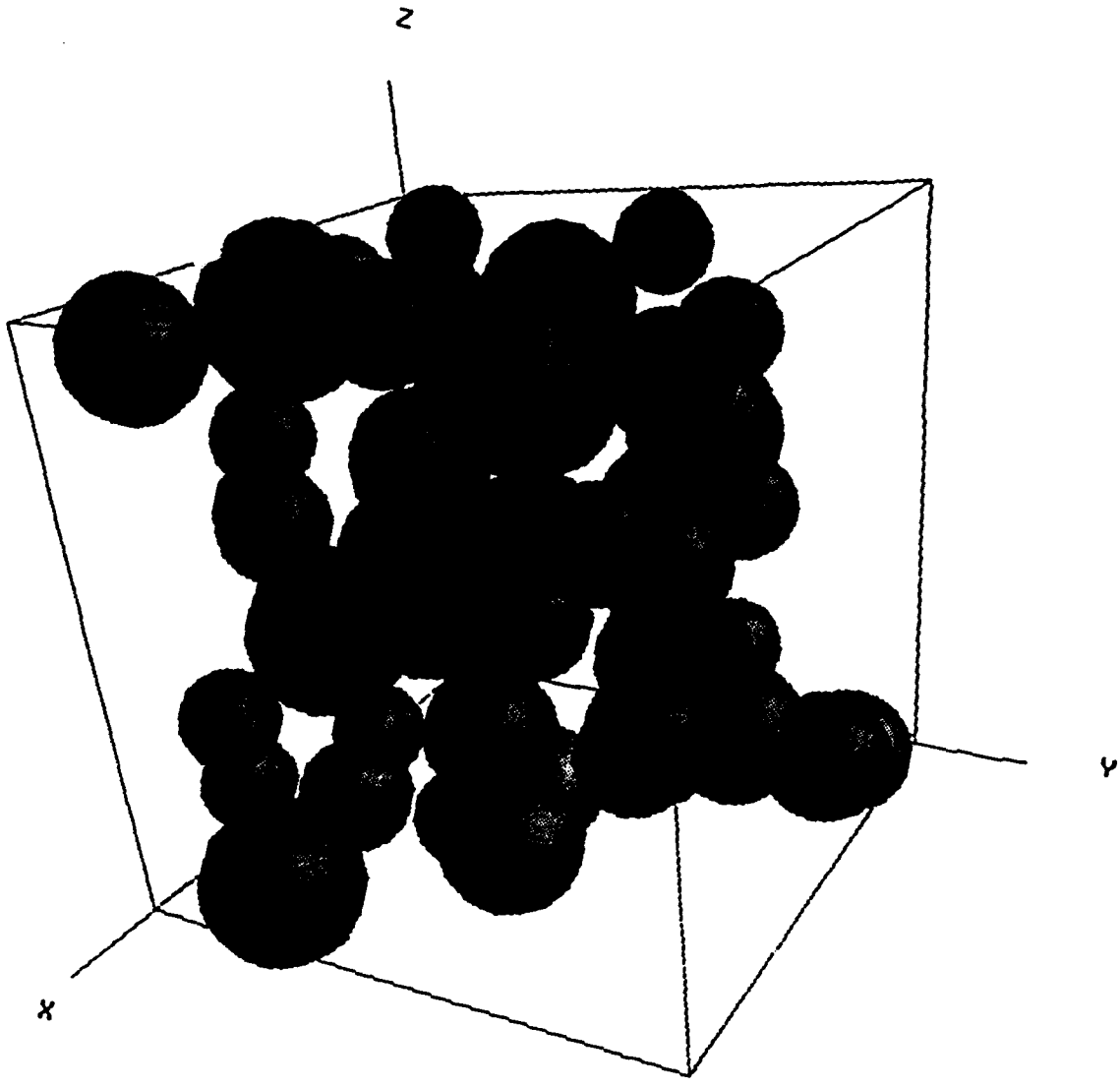


Figure 3.8: Random loose-packed configuration for 48 poly-disperse spheres (density=0.30).

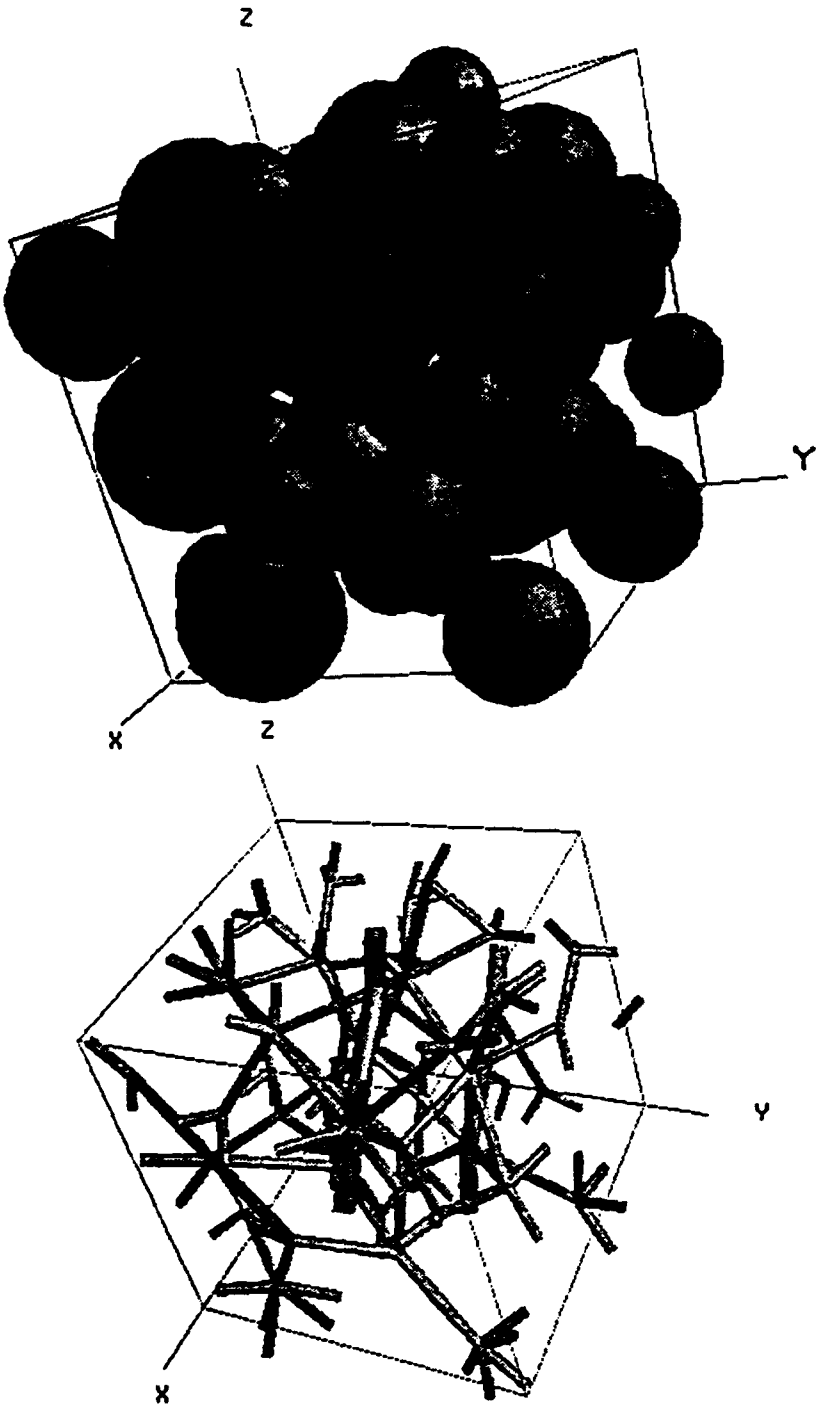


Figure 3.9: Random dense-packed configuration and associated contact bond network for 48 poly-disperse spheres(density=0.60). The thickness of rods represents the scaled magnitude of normal force between particles.

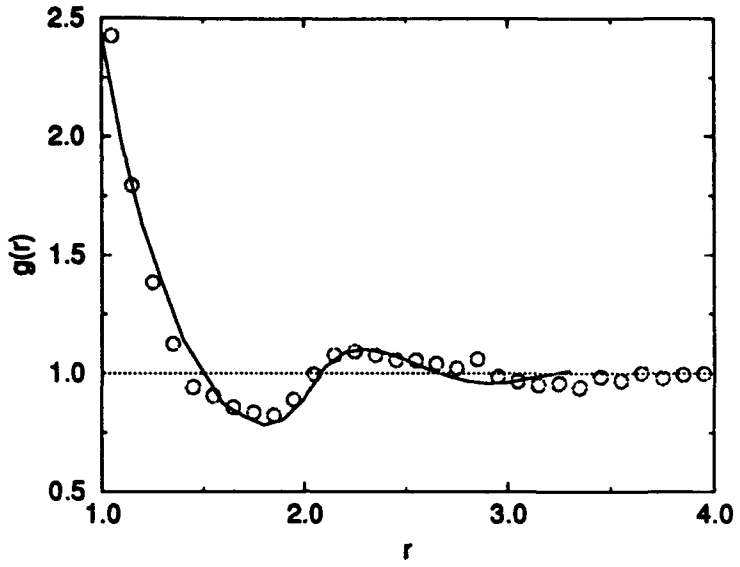


Figure 3.10: The simulated radial distribution function for 2D loose-packed configurations of 132 disks(discrete points), compared with the P-Y radial distribution function(solid curve)

those for 2D random dense packing). For 3D, Fig. 3.12 shows the smoothed $g(r)$ for 30 realizations of loose-packed configurations of 132 spheres(density=0.27). Finally, Fig. 3.13 shows $g(r)$ for 10 realizations of moderately dense-packed configurations of 90 spheres(density=0.58). Our computed radial distribution functions reveal that both the initially loose-packed and the final dense-packed systems are quite random, at least if one accepts the molecular model as a standard.

The text of this chapter, in part, appeared in [58]. The dissertation author was the secondary author of the publication, and shared equal responsibility with other co-authors.

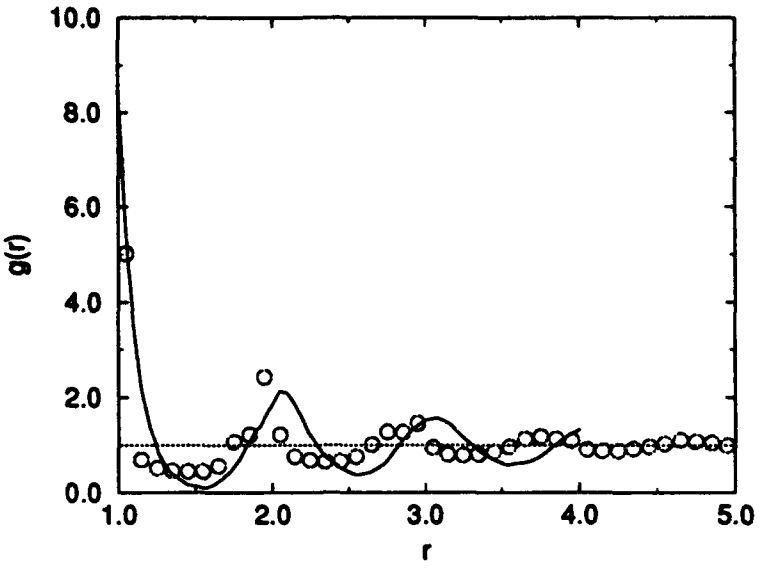


Figure 3.11: The simulated radial distribution function for 2D dense-packed configurations of 132 disks(discrete points), compared with the M-C radial distribution function(solid curve)

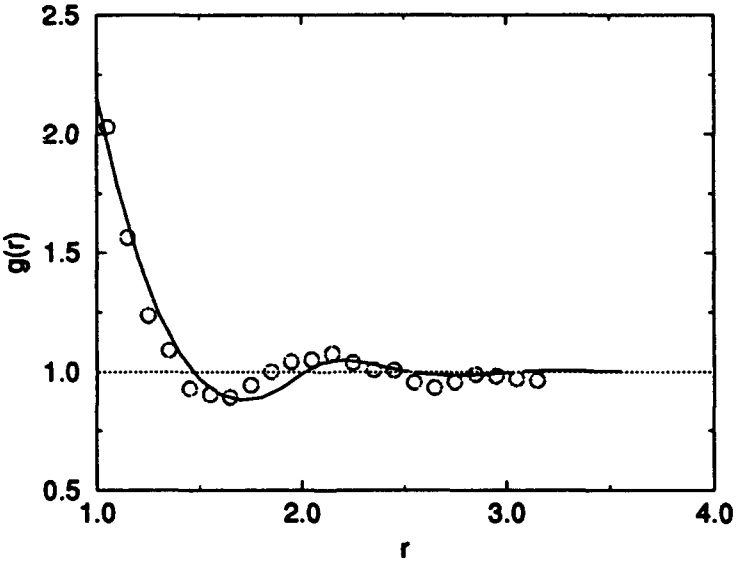


Figure 3.12: The simulated radial distribution function for 3D loose-packed configurations of 132 spheres(discrete points), compared with the P-Y radial distribution function(solid curve)

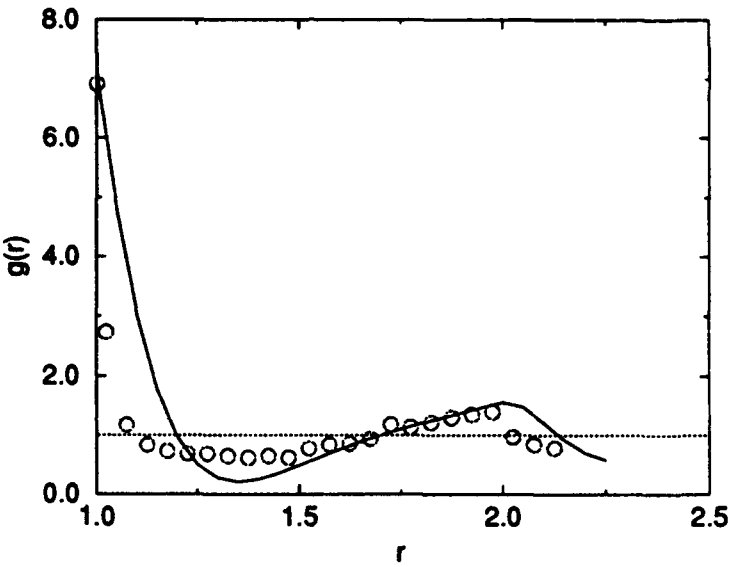


Figure 3.13: The simulated radial distribution function for 3D moderately dense-packed configurations of 90 spheres(discrete points), compared with the P-Y radial distribution function(solid curve)

Chapter 4

Experimental Investigation

The following sections describe the triaxial compression experiment employing steel balls as conductive granular particles.

4.1 Equipment

4.1.1 Compression Tester

The current experiments employ a commercial compression tester as loading frame for the test cell. The 911 MTT-02/10 compression tester, manufactured by Comten Industries Inc., is a motorized device with a digital interface and a variable speed drive. It consists of two parts, the main unit and a monitor/controller. The loading force is measured and displayed on the monitor/controller. The displacement of a specimen is measured with a separate 500 DC-E LVDT linear displacement transducer manufactured by Lucas-Schaevitz.

4.1.2 Triaxial Cell

The triaxial cell, a redesigned version of standard commercial cell with additional provision for conductivity measurement, is schematically illustrated in Figure 4.1. Furthermore, the cell dimensions have been reduced in order to fit into the

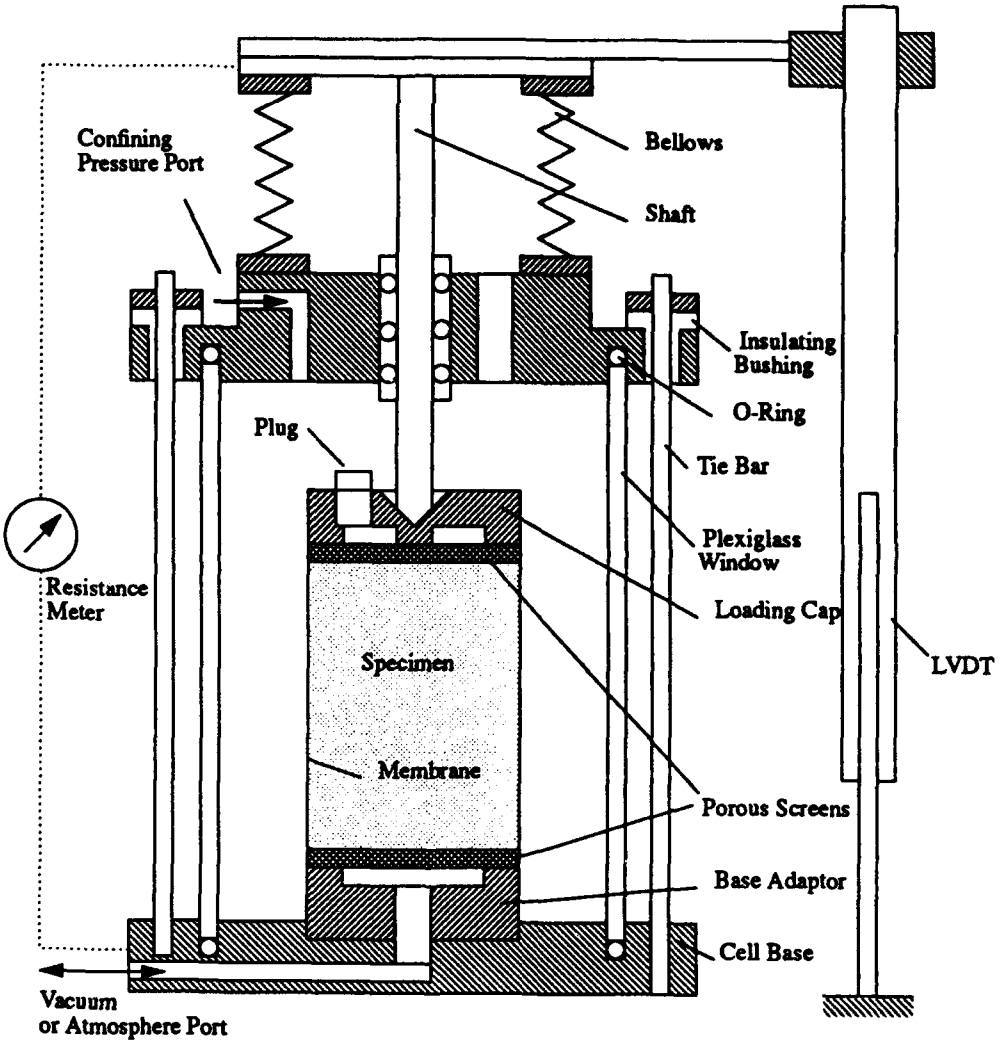


Figure 4.1: Schematic illustration of the triaxial cell.

compression tester, and a bellows is added to the top cap to prevent escape of confining fluid and to eliminate the friction between piston (shaft) and its bushing, a problem that is inevitable in the conventional design of the triaxial cells. The cell is made of stainless steel, with a transparent plexiglass cylindrical body. The base adaptor, with a porous steel screen installed on the top, has a channel for connection through the cell base to a vacuum system, for initial specimen preparation, or to the atmosphere for drained tests. The loading cap has also a porous steel screen at the bottom and an eccentric, small-diameter hole on the top to release air during the preparation of saturated specimens (not used in the current experiments). This small-diameter hole is always blocked by the threaded plug during experiments to prevent compressed air from escaping through the specimen. In the center of the loading cap, there is a conical recess, into which the end of loading shaft seats. In the top cap, there is an outlet from which the compressed air enters the cell chamber to form a confining pressure around the specimen. The LVDT device is mounted on the top cap to measure the end displacement of the specimen. The external connection to measure the resistance through the granular specimen is also shown in Fig. 4.1.

4.1.3 Digital Image Processing System

A PC-based digital image acquisition/processing system, used in a previous study [140], is employed for data acquisition. The hardware consists of a Hitachi KVC-150 video camera, a Toshiba M-6100 VCR, a Sony PVM-1271Q monitor, a IBM-PC compatible and an embedded PCVISIONplus PFGPLUS-512-3-u-XT/AT frame grabber, manufactured by Image Technology Inc.. The software includes a Werner-Frei Associates Image Lab and Imagetool programs.

The basic components of the system and their mutual relation are depicted in Figure 4.2.

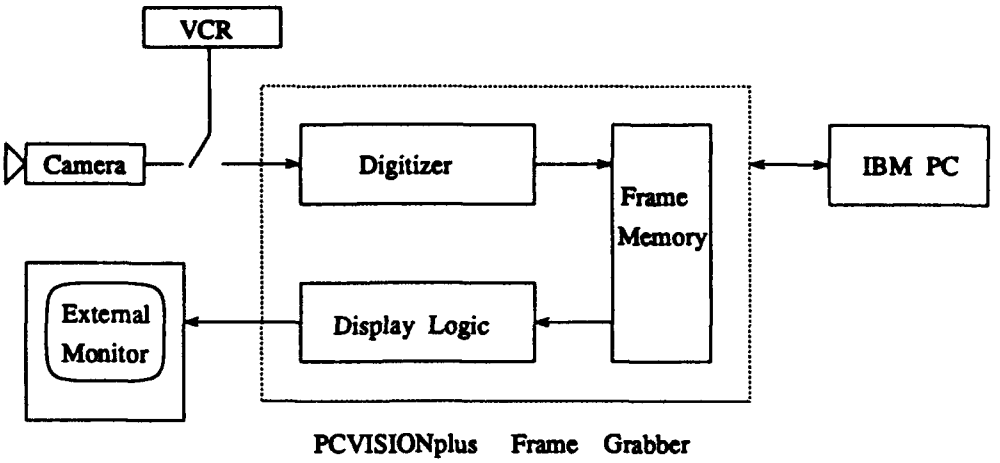


Figure 4.2: Digital acquisition/processing system, (after Zhuang 1991).

4.2 Materials and Specimens

The test material used as particles in the triaxial compression experiments is 440-C stainless steel balls provided by Thomson Precision Ball Company, Inc., the physical properties being listed in Table 4.1 in which the first four properties are provided by the manufacturer, the fifth is based on our own measurement, and the last one is found from CRC Handbook of the tables for Applied Engineering Science [21]. An interparticle friction coefficient $\mu = 0.15$ was measured by gluing the steel balls to a plate and observing the critical angle of sliding down a second inclined stainless steel plate. The granular assemblages consisted of a randomly packed beds of approximate 3400 of the above steel balls having a 3.81cm diameter and a 4.9cm height. The initial packing densities were around 0.6, which is close to that of a random dense packing of spheres. The ambient confining pressure is kept constant at $0.48 kg/cm^2$ throughout the course of the deformation.

Diameter (cm)	0.278 ± 0.000064
Density (g/cm^3)	7.667
Elastic modulus (kg/cm^2)	2.039×10^6
Poisson ratio	0.29
Friction coefficient	0.15
Electrical conductivity $(ohm \cdot cm)^{-1}$	16670.0

Table 4.1: The physical properties of the steel balls employed in the experiments.

4.3 Specimen Preparation and Experimental Procedure

First, the latex membrane is attached to the base adaptor, then stretched to the top of the split-cylinder, two-piece mold to form a cylindrical space. The steel balls are poured into the mold space in five portions, each is followed by rod-stirring, a procedure known as "rodding" in the soil-mechanics community, in order to achieve a consistent initial packing condition. The loading cap is then placed on the specimen and held by the membrane. A vacuum is then applied to withdraw the air from the specimen, which makes the specimen rigid under the atmospheric pressure. After the two-piece mold is removed, the plexiglass cylindrical body together with the top cap is installed on the cell base and tightened with three tie bars. The cell is then placed between two platens of the compression tester. The LVDT transducer is mounted on the top cap of the cell. Laboratory compressed air is used to fill an approximate $0.05m^3$ tank serving as air reservoir, which maintains a constant confining pressure during the experiments. Simultaneously, a confining pressure is created inside the triaxial cell that is always connected to the air reservior. The compressed air is shut off when the desired pressure p is reached. The vacuum is then turned off, and a valve is opened to vent the air in the specimen to accommodate drained tests. The specimen preparation is now complete.

Following the above specimen preparation, the compression tester is turned

on. The wires are connected between the cell and one electrical multimeter, which is utilized to measure the electrical resistance through the specimen. The LVDT is connected to a second multimeter, which measures the voltage output from the LVDT, thereafter being converted to the axial displacement. The video camera and VCR are set up to record the display readings of the monitor/controller and two multimeters. Everything is then ready for the experiment.

VCR recording and the compression tester are then started. The loading speed is slowly increased from zero to approximately 0.1cm/min. This loading process is terminated at 20% axial compressional strain.

4.4 Data Analysis

With the help of the image processing system, about 30 select frames are extracted from the recorded video image of the entire experiment at different stages, and the force, resistance and voltage readings are read off for later data analysis.

The measured force, corrected for the bellows spring force, gives us the axial force exerted on the specimen by the shaft. Then we compute the vertical stress over the specimen according to:

$$\sigma_1 = p + \frac{F}{A_{cap}} \tag{4.1}$$

where p is the confining pressure, F is the axial force exerted on the specimen by the shaft, and A_{cap} is the area of the loading cap. The horizontal stresses σ_2 and σ_3 are equal to the confining pressure p.

According to the definition, we obtain

$$K^* = \frac{L}{R_t A_s}. \tag{4.2}$$

where R_t is the total resistance through the specimen, K^* is the effective conductivity of the medium, L is the specimen length and A_s is the cross-section area of the specimen.

4.5 Measurements of Contact Resistance

Faced with the extremely large discrepancy between the experimental measurements and the results of the initial numerical simulations utilizing eq. 2.23 to compute the contact resistance, we were lead to identify the underlying cause. As a matter of fact, there has been ongoing research into electric contact resistance for the past several decades [64,60,61,6,125,68,83,24,119,25], since almost all electrical or electronic equipment contains numerous contacts through which the current and voltage signals are transmitted, and the failure of even a single contact can result in a complete system failure. From the above literature survey, one finds that the resistance at a real contact may be much higher owing to the various reasons outlined in the section 2.3.2. This ultimately results in lower effective conductivity of granular medium than that predicted by the theory.

Based on this understanding, we conducted an experiment to measure the electrical contact resistance in a single column of steel balls as a function of axial load. The experimental set-up is schematically depicted in figure 4.3. A stack of balls is confined within a reinforced quartz glass capillary tube, with an inner diameter 0.29cm, slightly larger than that of the steel balls, and loaded with dead weight from the top via a piston. The resistance is measured with a multimeter at various loads. To avoid all contacts between the balls and flat surfaces, the top and bottom steel balls in the column, consisting of five balls, are soldered to the piston and the flat base, respectively. The relationship between resistance and load, the average result of seven experiments, is plotted in figure 4.4 and compared with that predicted by Eq. 2.23. The overall scatter is within 20% of the average values. The large scatter is probably ascribed to the nonuniformity of the ball surfaces and associated contamination films. One can see a very strong dependence of resistance on the load, with a slope of approximately 2.4 compared with 1/3 given by Hertzian theory. This observation suggests that an oxide film on the ball surface ruptures and deforms plastically under the applied load.

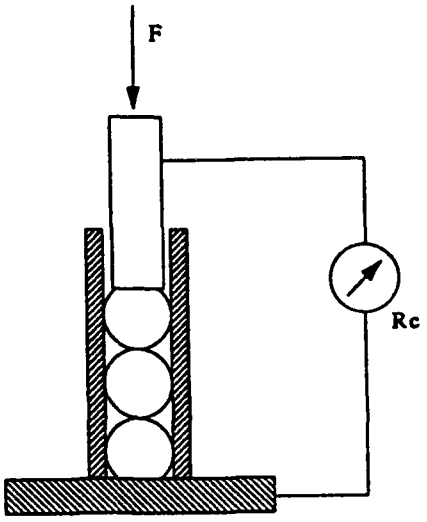


Figure 4.3: Experimental set-up used to measure the electrical contact resistance.

The load-resistance curve depicted in Fig. 4.4 represents the case of loading. Experimental observations revealed that the load-resistance curve for unloading is lower than that of loading. This hysteresis is probably due to the plastic deformation of asperities and irreversible rupture of the superficial oxide film during loading. While the contact between two balls can experience loading and unloading during the deformation of granular media, the localized contact regions on the ball surfaces keep changing due to the relative movement (rolling and sliding) between balls. Therefore, it is plausible to employ the normal load–contact resistance relation of Fig. 4.4 for loading for purposes of numerical simulation.

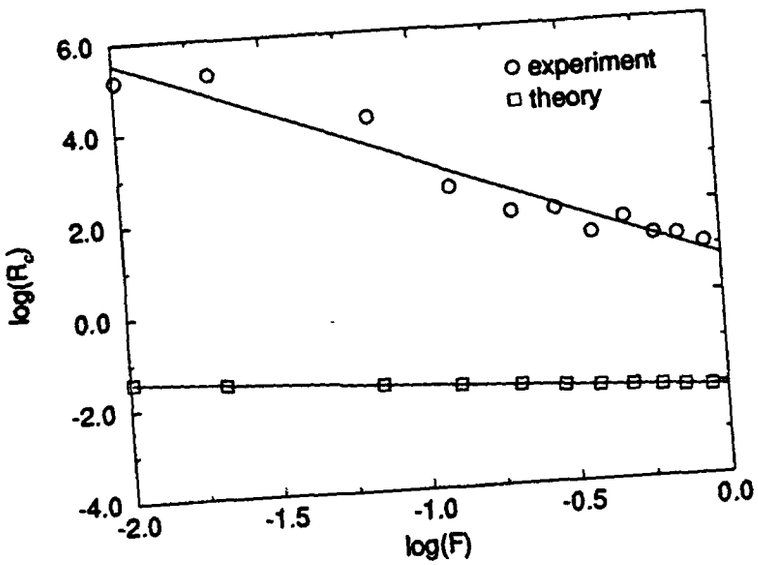


Figure 4.4: Relation between contact resistance and normal load: comparison of the measured contact resistance with the theoretical Hertzian-contact prediction.

Chapter 5

Simulation Results

A series of numerical simulations on idealized granular assemblages have been conducted to investigate the effects of microscopic and microstructural properties of the constituent particles and their assemblages, including (Coulomb) interparticle friction, nonlinear contact mechanics and initial packing density, on the mechanical behavior. Of particular interest is the Reynolds dilatancy, shear strength and the evolution of granular microstructure of idealized granular assemblages subject to constant mean confining pressure. Numerical simulations of the triaxial compression test have been conducted to simulate the effects of the initial density on the mechanical behavior as well as the scalar transport properties. The mechanical behavior is discussed first in this chapter, while transport properties will be covered separately in the following chapter.

5.1 Interparticle Friction

This study involves both 2D and 3D mono- and poly-disperse granular assemblages subject to simple shear deformation under constant mean confining pressure. The 2D assemblages consist of 132 disks initially packed to random dense packing with about 0.82 area fraction. On the other hand, the 3D granular assemblages contain 48 spheres initially packed to an approximate dense random packing

with 0.60 volume fraction. The nondimensional radii of the particles are equal to 1.0 for monodisperse systems, and equal to 0.8, 1.0, and 1.25, respectively, in the polydisperse systems, with approximately same total volume of particles of the three different sizes. Thus, we use 64 disks of radius 0.8, 41 disks of radius 1.0 and 27 disks of radius 1.25 for 2D systems, and choose 27 spheres of radius 0.8, 14 spheres of radius 1.0 and 7 spheres of radius 1.25 for 3D systems. Normal and shear contact stiffnesses are taken to be $k_n = 1.0$ and $k_t = 0.8$, respectively.

By a scaling based on contact stiffness and particle radius, one can specify an externally imposed nondimensional pressure ($=p_r R_r/k_{n_r}$, where p_r, k_{n_r} and R_r denote the real confining pressure, normal stiffness and particle radius.), under which interparticle overlap(proportional to normal force) will not exceed 0.1% of particle radius throughout the simulations since we are primarily concerned with nearly rigid particles [14]. This pressure is found to be 6.0×10^{-5} for 2D experiments and 4.0×10^{-5} for 3D experiments, and is maintained during the initial packing process and subsequent shearing. Both the 2D or 3D test assemblages are subjected to simple shear up to 20% total strain with different interparticle friction coefficients under otherwise identical conditions.

To further clarify the influence of particle friction on Reynolds dilatancy of randomly dense-packed granular a ssemblages, we carried out several simulations on both 2D and 3D, and mono- as well as poly-disperse idealized granular assemblages with μ=0.0, 0.3, and 0.5, respectively. The following conclusions can be drawn from the results, presented in Figure. 5.1, 5.2, 5.3,and 5.4: the dilatancy increase with increasing magnitude of μ, which is in agreement with previous results [39,18,33], including the results for polydisperse (random-packed) cases found by Bashir and Goddard[14]. However, this finding is contrary to Reynolds' original hypothesis on the random dense packing of granular assemblages, as interpreted. The stress ratio $(\sigma_1 - \sigma_3)/p$, where σ_1, σ_3 and p are major, minor and mean stress, also increases with increasing magnitude of μ. Polydispersity is found to have a noticeable effect on the mechanical behavior of granular assemblages.

From Fig. 5.5 and 5.6, one finds that the coordination number decreased drastically in the start of shearing, usually within 1% of shear strain, which indicates that a significant particle rearrangement took place early in the shearing deformation [79,95,96]. The coordination number then fluctuated about a roughly constant level throughout the subsequent deformation [18]. A higher interparticle friction generally results in a lower coordination number after granular assemblages yield. Although the computed coordination number varies with shear strain and interparticle friction, it is found to be, in both 2D and 3D cases, always greater than the geometric critical coordination number $Z_c \approx \frac{d}{d-1}$ (= 2 for $d = 2$ and 1.5 for $d = 3$, here, d, > 1, denotes the number of dimensions [116]). Z_c, at which there exists an infinite cluster of bonds across a medium according to percolation theory [116,75], is a dimensional invariant insensitive to the details of the lattice studied. The geometric percolation threshold p_c is shown to be about 0.347 for 2D triangular lattice and 0.119 for FCC lattice [116], while the elastic (central-force, omitted hereafter) bond percolation threshold p_{cen}, which would associate with solid behavior of bond networks at small strain, is found to roughly equal to 0.58 for 2D triangular lattice and 0.42 for FCC lattice [49]. Here, we define a ratio Z/Z_{max} representing the fraction of active "bonds" in the network of particle contacts compared with the coordination number of the densest possible systems, where Z being the coordination number of a system and Z_{max} the maximum possible coordination number, for instance, 6 for 2D triangular lattice and 12 for 3D FCC lattice. In 2D monodisperse case, from Fig. 5.5, one finds a ratio 0.6 at initial stage, slightly larger than the elastic bond percolation threshold, and a range from 0.4 to 0.5, dependent on interparticle friction, after early shearing deformation, which is between the geometric and elastic percolation threshold. Similar results are found in 3D monodisperse case. In a previous 2D work, however, Bashir and Goddard [14] found the ratio very close to the geometric percolation threshold after initial 3% shear strain, which is much smaller than the elastic percolation threshold.

Based on detailed microscopic observations, we find that: (1) granular microstructure evolves such that contact normals concentrated in the direction of major

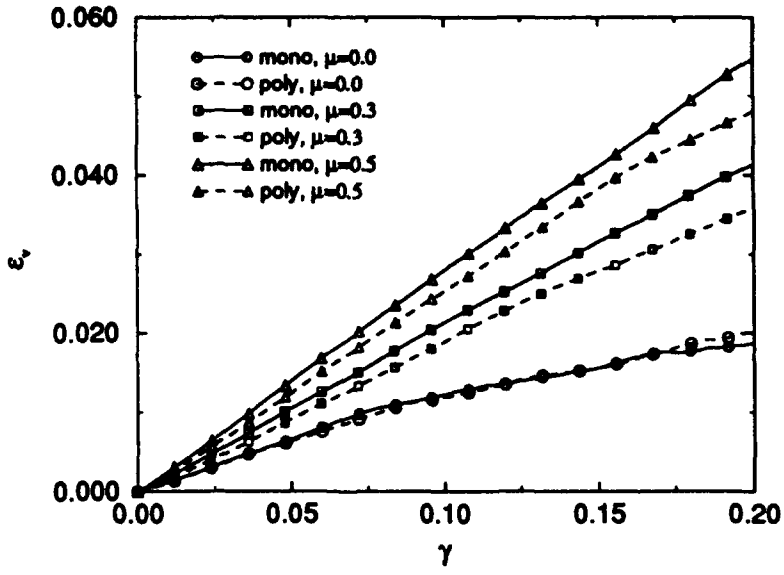


Figure 5.1: Effects of interparticle friction coefficients on dilatancy of 2D assemblages subjected to simple shearing deformation.

principal stress during the shearing deformation [124]; (2) the granular assemblage is composed of two types of region: a major skeleton composed of heavily stressed chains of particles and less stressed regions surrounding this skeleton, with most of contact breaking and making occurring within the latter region and with the skeleton remaining relatively unaltered for a small incremental deformations; (3) particle rolling is major deformation mechanism, especially when interparticle friction is large[102].

5.2 Nonlinear Contact Mechanics

As we discussed in section 2.2, the contact stiffness is generally a function of load, often well represented by the power law

$$k_n = C f_n^{\lambda} \tag{5.1}$$

where C is a material constant, f_n normal load, and for example, the exponent takes on values $\lambda = 1/3$ for Hertzian elastic spheres and $\lambda = 1/2$ for a conical tip pressed

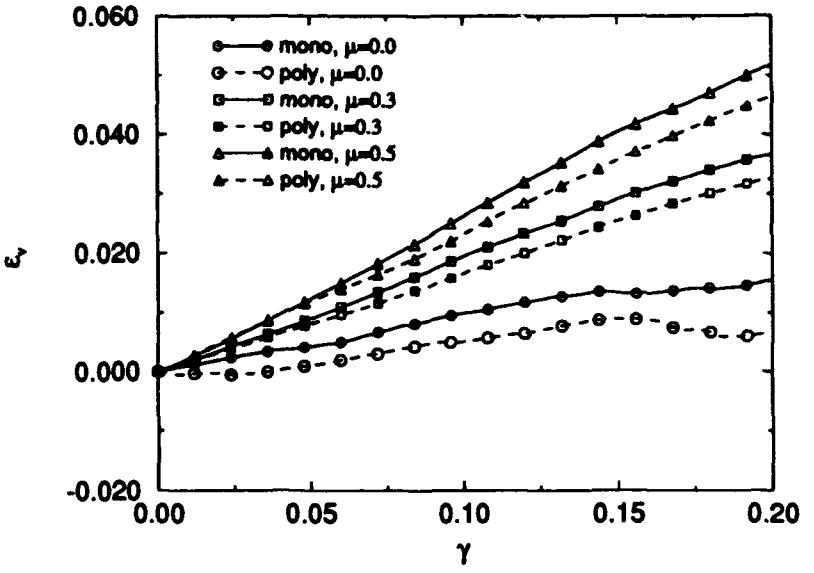


Figure 5.2: Effects of interparticle friction coefficients on dilatancy of 3D assemblies subjected to simple shearing deformation.

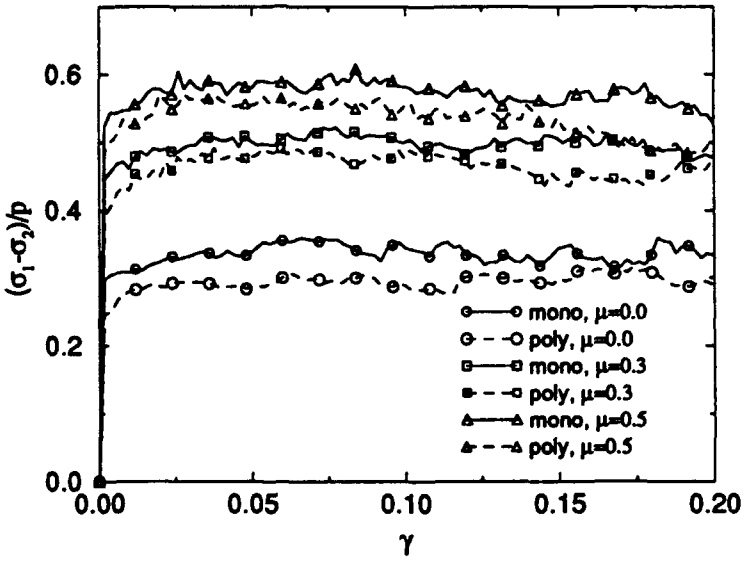


Figure 5.3: Effects of interparticle friction coefficients on shear strength of 2D assemblages subjected to simple shearing deformation.

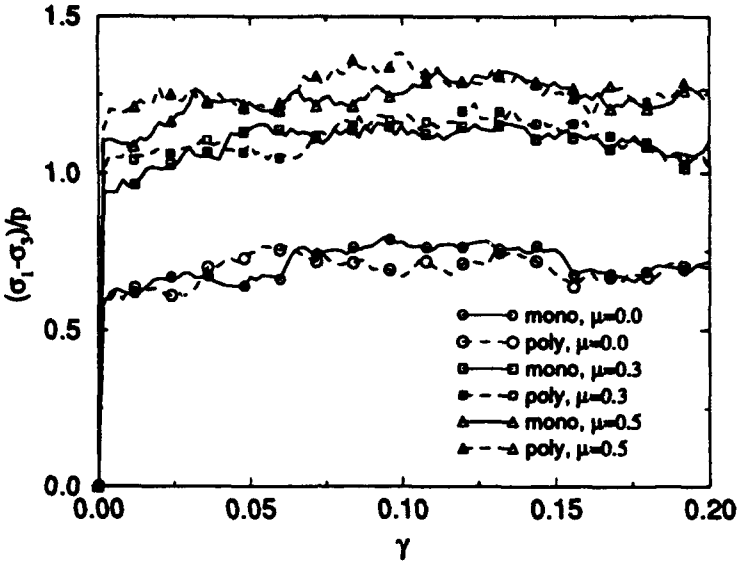


Figure 5.4: Effects of interparticle friction coefficients on shear strength of 3D assemblages subjected to simple shearing deformation.

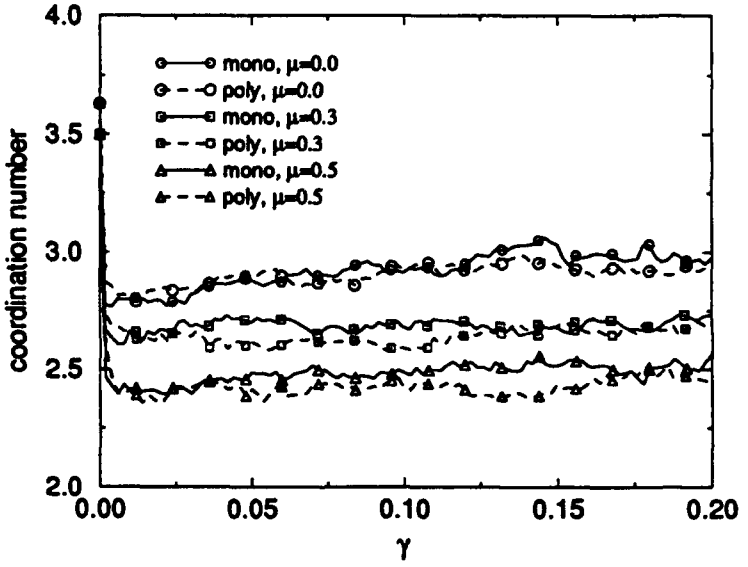


Figure 5.5: Effects of interparticle friction coefficients on average coordination number of 2D assemblages subjected to simple shearing deformation.

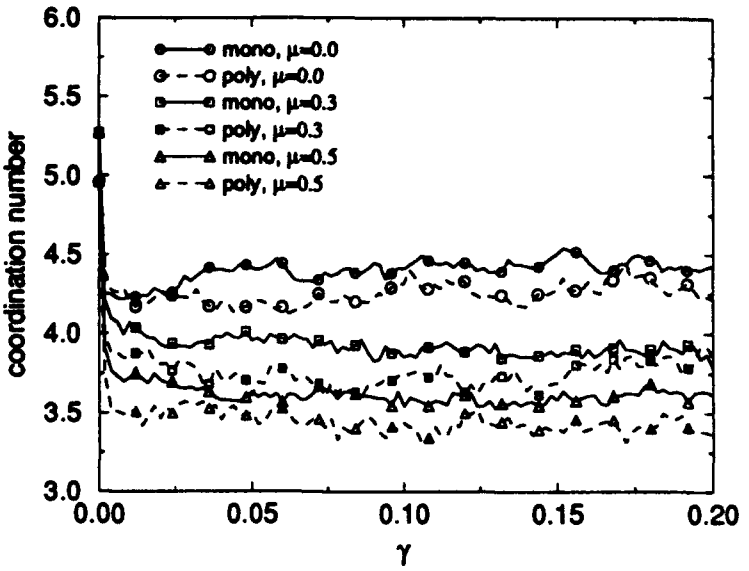


Figure 5.6: Effects of interparticle friction coefficients on average coordination number of 3D assemblages subjected to simple shearing deformation.

against a plane [57]. Accordingly, the relationship 3.3 between increments of relative displacement and contact force is nonlinear.

Since the linear model $\lambda = 0$ not only offers simplicity but may be able to provide qualitatively valid insights into the link between micromechanical properties and macroscopic behavior [17,18,39,40,41,128] and, since we are mainly interested in ideal rigid limit [14], most of our simulations have been carried out for linear contacts. However, we felt that it is important to check its validity and the effects of nonlinearity.

In the present simulations we employ a monodisperse system with 48 spheres packed to initial density $\phi = 0.60$ and interparticle friction $\mu = 0.15$. The system is subjected to the triaxial cor· ·ssion under a constant mean confining pressure $p = 4 \times 10^{-5}$. λ values are selected as 0.0, 1/3, 1/2, and 1, with 0.0 representing the linear contact and 1.0 representing extreme nonlinearity. The tangential stiffness k_t is simply taken to be $0.8k_n$.

From Fig. 5.7, one sees that contact nonlinearity has no apparent influence

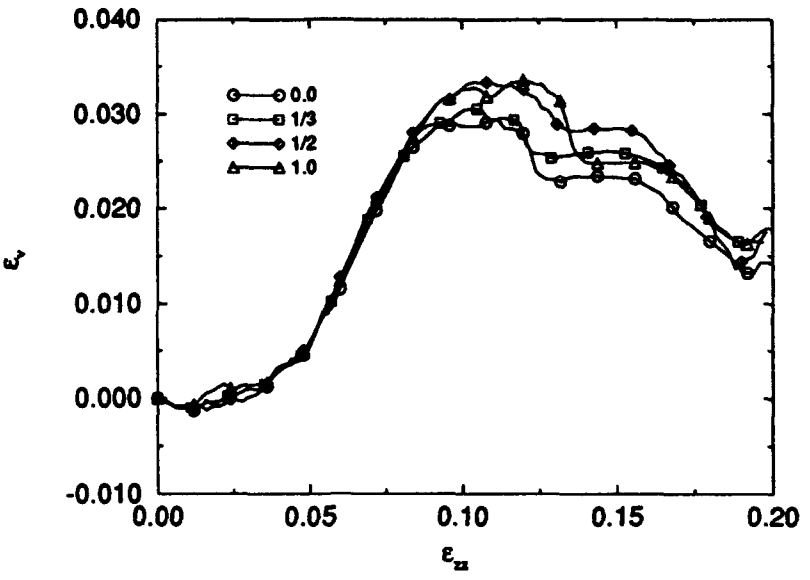


Figure 5.7: Effects of non-linear contact on dilatancy of 3D assemblages subjected to triaxial compression.

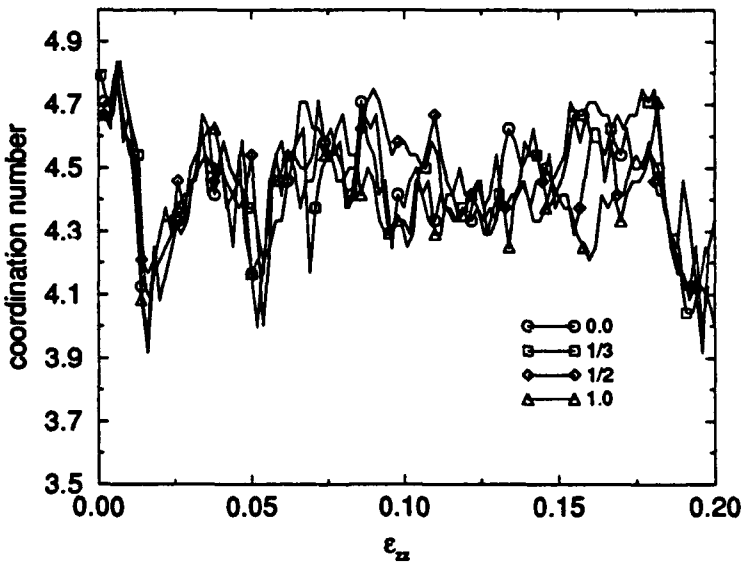


Figure 5.8: Effects of non-linear contact on average coordination number of 3D assemblages subjected to triaxial compression.

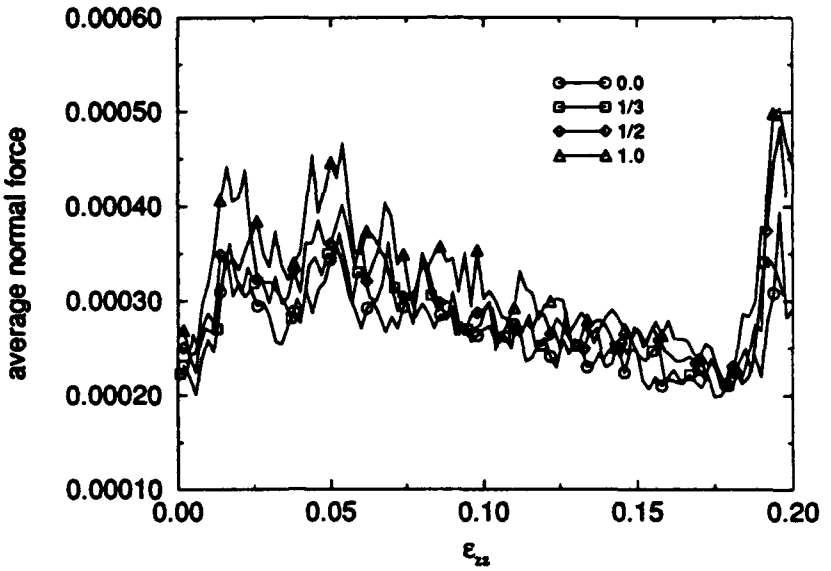


Figure 5.9: Effects of non-linear contact on average contact normal force f_n in 3D assemblages subjected to triaxial compression.

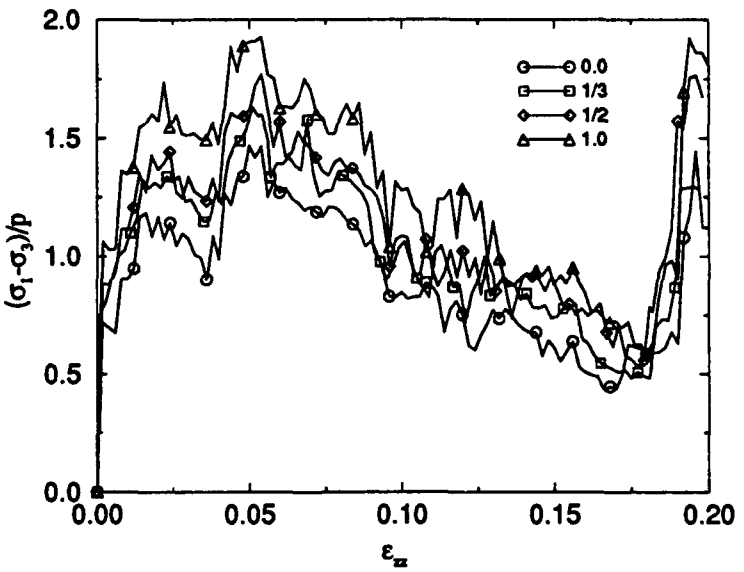


Figure 5.10: Effects of non-linear contact on shear strength of 3D assemblages subjected to triaxial compression.

on Reynolds dilatancy in the small- to intermediate-strain region, but some effects are observed at higher axial strains. Moreover, Fig. 5.8 indicates that the average number of particle contacts is, roughly speaking, not affected. It is expected that the nonlinear contact law will tend to make the strong contacts (in terms of contact force) stronger, and the weak contacts weaker, therefore, changing the force distribution. Although we do indeed observe variations in the force distributions, the different degrees of nonlinearity yield only a small deviation in the average normal force, for the cases $\lambda = 0.0$, $1/3$ and $1/2$ (see Fig. 5.9). Finally, from the results in Fig. 5.10, one sees that the strength of the idealized granular assemblages tends to increase somewhat with the increase of nonlinearity. The small effects of contact nonlinearity on the mechanical behavior indicate that a complicated nonlinear contact law is less significant in modeling the mechanical behavior of granular materials.

5.3 Effects of Initial Specimen Density

To simulate the effects of the initial void ratio or density on the behavior of mechanical as well as the scalar transport properties, we have generated three random monodisperse packings of 100 spheres with different initial densities, 0.52, 0.56 and 0.60, respectively. The interparticle friction coefficient μ is taken to be 0.15. All three packings are subjected to the triaxial compression deformation under the same nondimensional confining pressure $p_0 = 4 \times 10^{-5}$ in the directions normal to compressional axis.

Fig. 5.11 and 5.12 indicate that, for initially loose systems such as those with $\phi = 0.52$ and 0.56, densification or negative dilatancy occurs initially and persists throughout the deformation. The potential for densification increases with the decrease of the initial density [91]. On the other hand, the initially dense system, with $\phi = 0.60$, experiences positive dilatancy from the very beginning of the deformation. Nevertheless, the densities of three systems, either contracting or expanding, tend to approach the same critical value asymptotically. Fig. 5.13 shows that, for

loose systems, the shear strength increases monotonically. However, for the initially dense system, its strength increases first until a peak value is reached, then decreases. Similar observations in real triaxial compression tests are to be discussed in following chapter. Again, both the loose and dense systems seem to possess an identical ultimate strength after being subjected to a large deformation [91]. The initial coordination number increases as the density rises. Upon deformation, the dense system initially experiences a significant loss in the number of contacts, whereas the loose system gains contacts. However, both systems approach approximately the same coordination number at roughly 1% axial compressional strain, and then maintain fluctuating but slightly increasing values (see Fig. 5.14). Next, I shall attempt to elucidate the observed behavior.

In a loose system, the initial number of interparticle contacts and coordination number are low, and just exceeds slightly the elastic-percolation threshold, at which there just begin to exist sample-spanning chains of particles capable of supporting an ambient confining stress (Goddard 1990). However, due to lack of sufficient contact force from neighboring particles, such chains are highly unstable to (Euler) buckling. Under these circumstances, one can anticipate that a given pair of adjacent particles in such a chain will accommodate axial compression with a small rotation normal to their line of centers until such rotation is hindered by lateral contact with neighboring particles. By means of this process, a given particle chain will generally undergo a kind of lateral 'branching' until it becomes capable of supporting increased axial compressive stress (Goddard 1990). Therefore, the overall granular structure is less stable and more likely to collapse to a more stable, denser system upon deformation and to generate load-bearing capability. Such capability is further enhanced as the system gets denser. On the other hand, the dense system will have to expand in order to deform, hence loses contacts initially. Owing to the volume expansion against the ambient confinement, the system exhibits shear strength, but further dilatancy reduces the system density and decreases the stability of the granular chain structure and its ability to support the external loads. This explains the after-peak strength

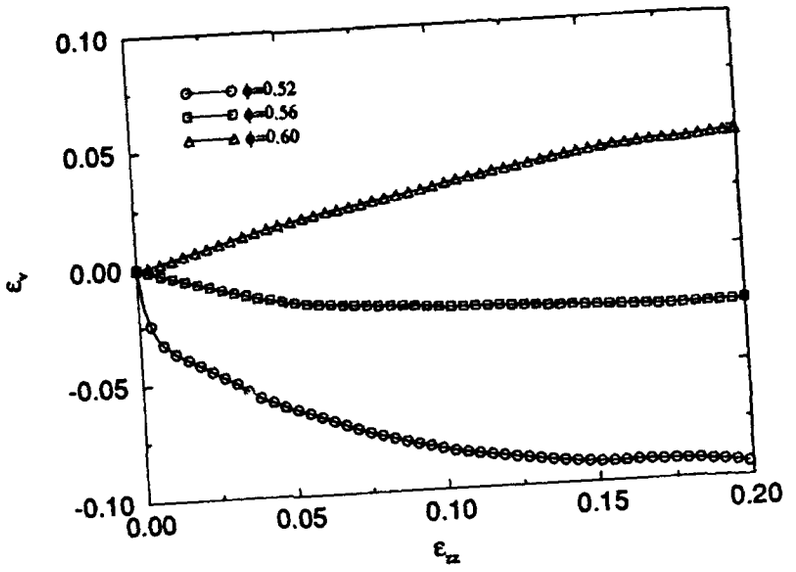


Figure 5.11: Effects of initial density on dilatancy of 3D assemblages subjected to triaxial compression.

loss for dense systems illustrated in Fig. 5.13.

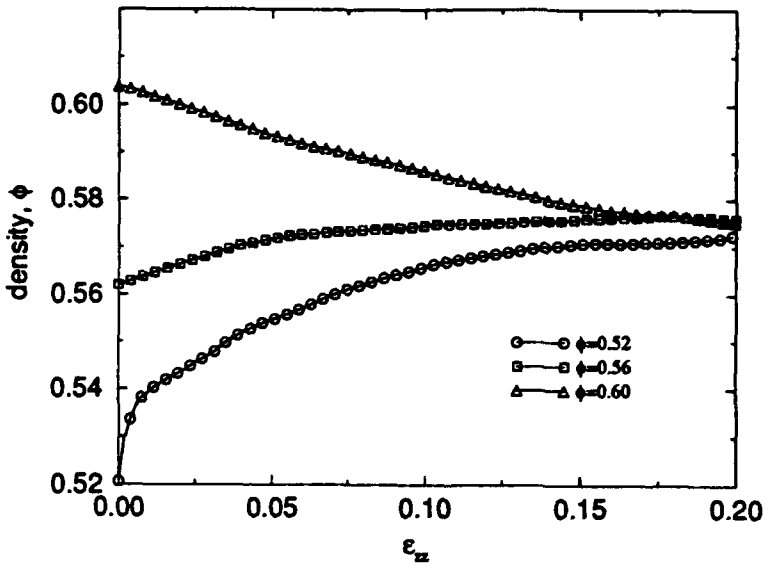


Figure 5.12: Effects of initial density on density evolutions of 3D assemblages subjected to triaxial compression.

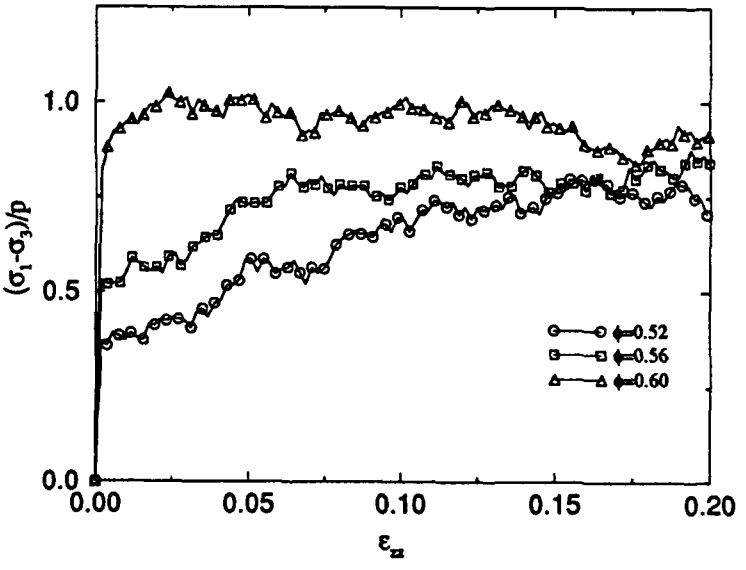


Figure 5.13: Effects of initial density on shear strength of 3D assemblages subjected to triaxial compression.

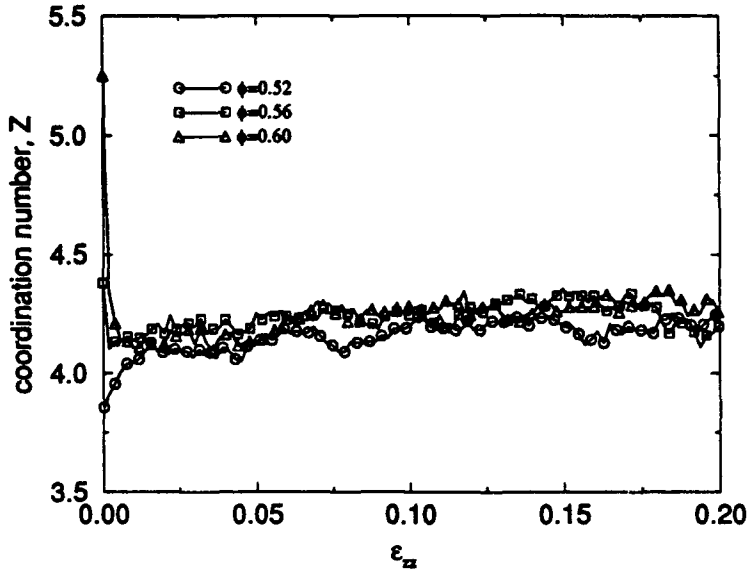


Figure 5.14: Effects of initial density on coordination number of 3D assemblages subjected to triaxial compression.

Chapter 6

Electrical Conductivity

6.1 Numerical Simulations

For the purpose of data validation, the computer code is modified specifically to simulate a triaxial compression experiment over an idealized granular assemblage with the physical properties of steel balls, listed in table 4.1, and experimental loading conditions as input parameters. The mechanical behavior as well as the electrical transport properties of the systems are investigated simultaneously.

The idealized system, as schematically depicted in figure 6.1, contains 100 equal size spherical particles in a periodic cubic cell. The system, being confined in X and Y directions with a constant pressure p_0, is compressed in Z direction. To simulate the effects of the initial density on the mechanical as well as scalar transport properties, we have generated three isotropic random packings of sphere assemblages with initial densities being 0.52, 0.56 and 0.60, respectively. The interparticle friction coefficient μ is taken to be 0.15, corresponding to the measurements on the real 'dirty' steel balls (whose definition is to be provided in Section 6.2). The computed dimensionless normal force is converted to real force by the scaling factor based on the particle radius and ambient confining pressure so that the contact resistance between two particles can be calculated with the experimentally determined load-resistance relation given in figure 4.4, also corresponding to the 'dirty' steel balls. The tangential

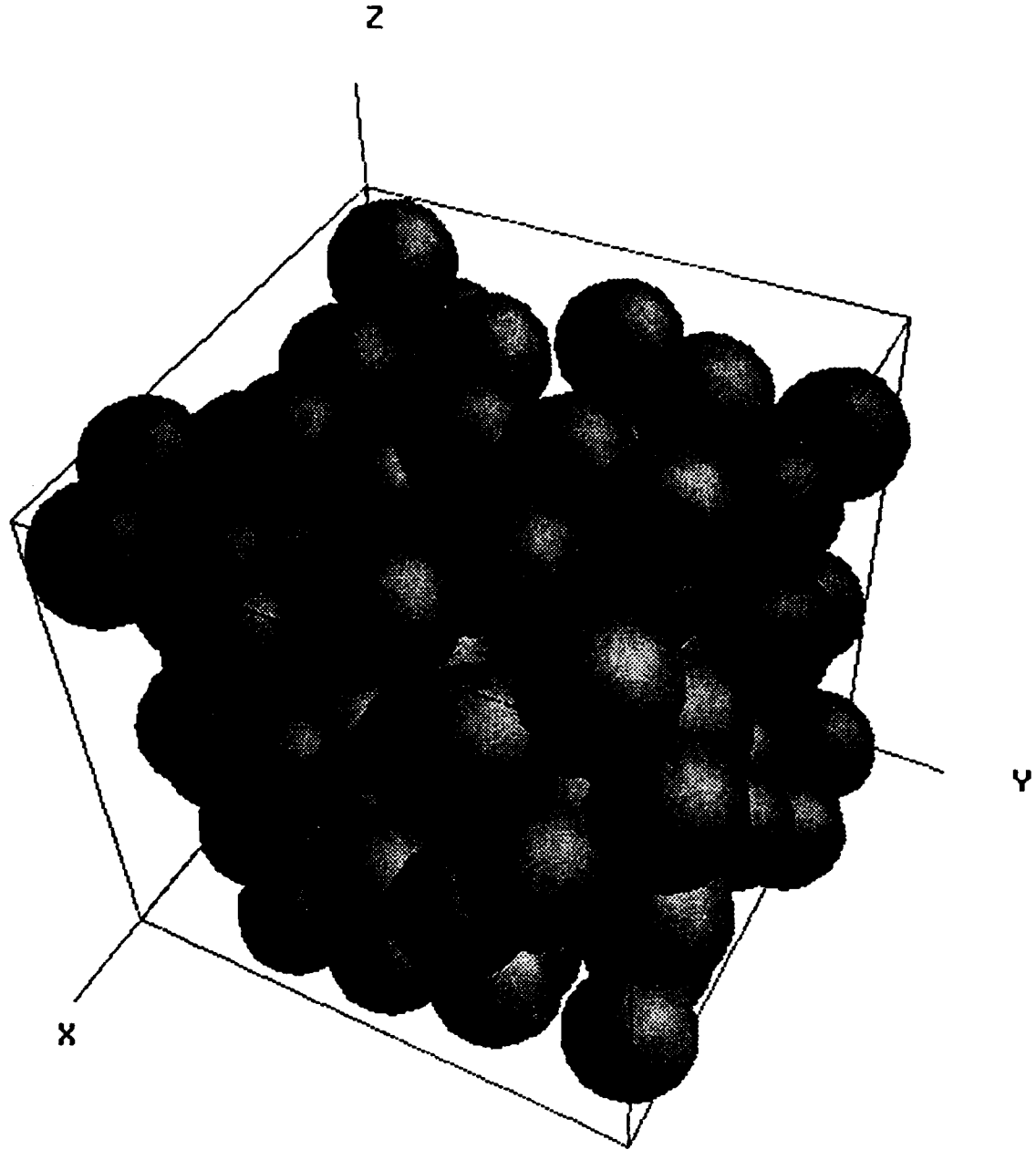


Figure 6.1: The idealized system in triaxial compression ($\phi = 0.60$).

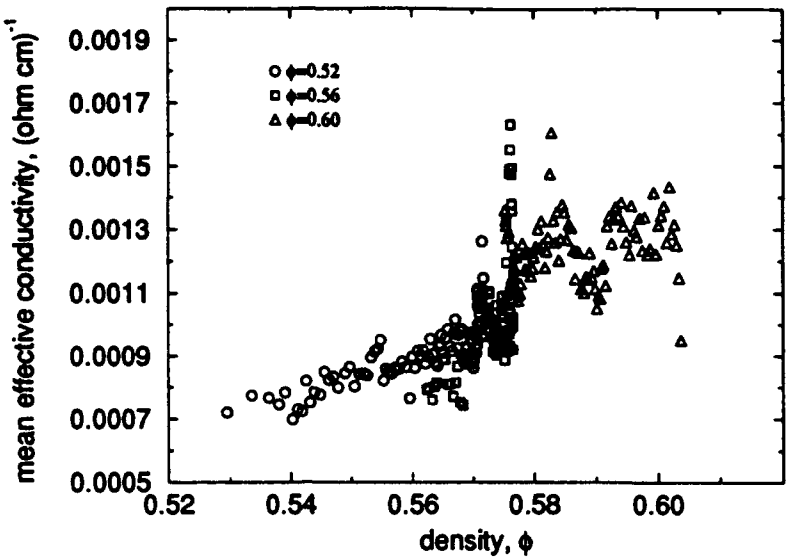


Figure 6.2: Relation between mean effective conductivity and specimen density.

force is not considered in evaluating the contact resistance, since, according to our experimental observations, the tangential force has no apparent effect on the contact resistance. The results reported in this section correspond to the physical properties of 'dirty' steel balls.

It is generally true that a denser system should possess higher conductivity or lower resistivity owing to the fact that medium is better connected so as to offer more paths or branches for current to pass through. Fig. 6.2, where the mean effective (the word 'effective' and corresponding superscript '*' is henceforth omitted) conductivity is the average of conductivities K_{xx}, K_{yy} and K_{zz} in X, Y and Z directions, reveals this trend although more fluctuation are observed in high density region.

Next, we shall consider the changes of the conductivity K_{zz} in the compressional (or Z) direction during the course of deformation. From Fig. 6.3 and 5.13, it is not difficult to see the similarity between the behavior of shear strength and conductivity, although there are more fluctuations in the conductivity than in strength. For the initially dense system, with $\phi = 0.60$, the conductivity increases first to a peak within the first 2% of axial strain, then fluctuates wildly with a decreasing trend.

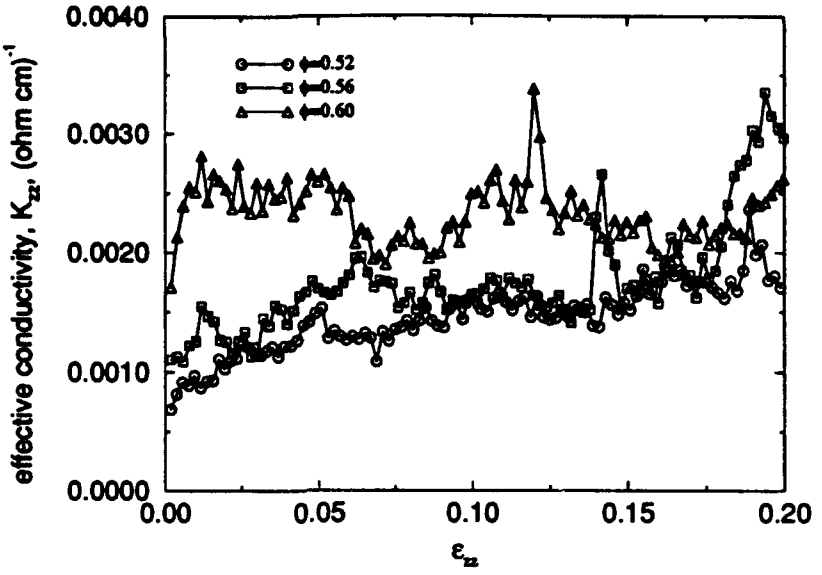


Figure 6.3: Dependence of conductivity on axial strain and initial density.

This behavior can be correlated to the mechanical behavior, particularly deformation mechanism of the granular assemblages.

Upon the compression, load-bearing chains are built up gradually in the Z direction in order to sustain stress anisotropy, which create easier paths for current to pass through. Therefore, in early stage of deformation, the conductivity increases steadily despite the fact that system experiences a loss in total number of contacts (see Fig. 5.14), note that the loss of contacts happens mainly on X-Y plane or the direction of minor principal stress. Owing to the dilatancy of the system (see Fig. 5.11), these granular chains become less stable progressively. When the system is further expanded, these major load-bearing chains finally buckle. The branching-out of chains diverts current from the preferred direction. Therefore, one observes the after-peak decrease in conductivity. The subsequent built-up and buckling of new heavily-stressed chains is primary cause of the fluctuations in conductivity. On the other hand, the densification in the loose systems during the deformation tends to stablize these progressively loaded chains, thus resulting in a steady increase of conductivity.

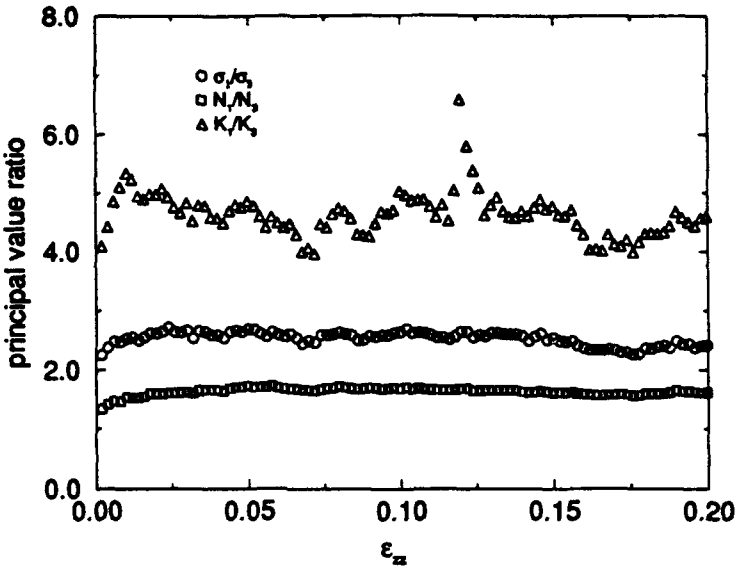


Figure 6.4: Evolution of principal value ratios for stress, fabric and conductivity ($\phi = 0.60$).

The principal value ratio, the ratio of major to minor principal value of tensors such as stress or fabric, is often used to characterize the state of anisotropy. Following is an attempt to correlate the evolution of the anisotropy of three tensors, namely stress $\boldsymbol{\sigma}$, fabric $\mathbf{N}$ and conductivity $\mathbf{K}$ tensors. At the start of compression, the anisotropy of the assemblages is induced progressively. After reaching a peak, it remains relatively unchanged throughout the deformation. From Fig. 6.4, one see fabric is not very sensitive to the change in stress state, while conductivity is more sensitive.

Fig. 6.5 and 6.6 clearly suggest strong correlations between conductivity, stress and fabric tensors. Note that the lines of linear regression almost pass through the point (1,1) in the plots, which represents the isotropic relation between these tensors.

The results shown in Fig. 6.7 indicate that the mean field theory of Batchelor and O'Brien underpredicts the effective conductivity slightly but can be used to understand the qualitative scalar transport behavior of granular materials.

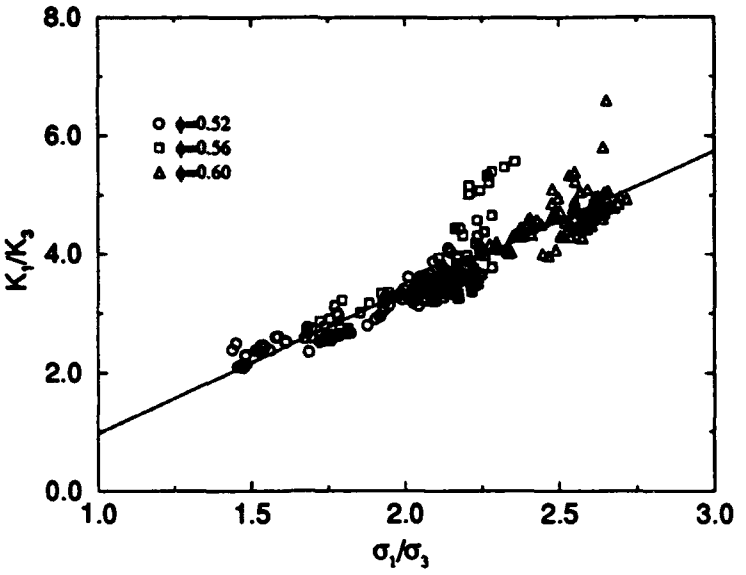


Figure 6.5: Relations between principal value ratios of conductivity and stress (slope=2.39).

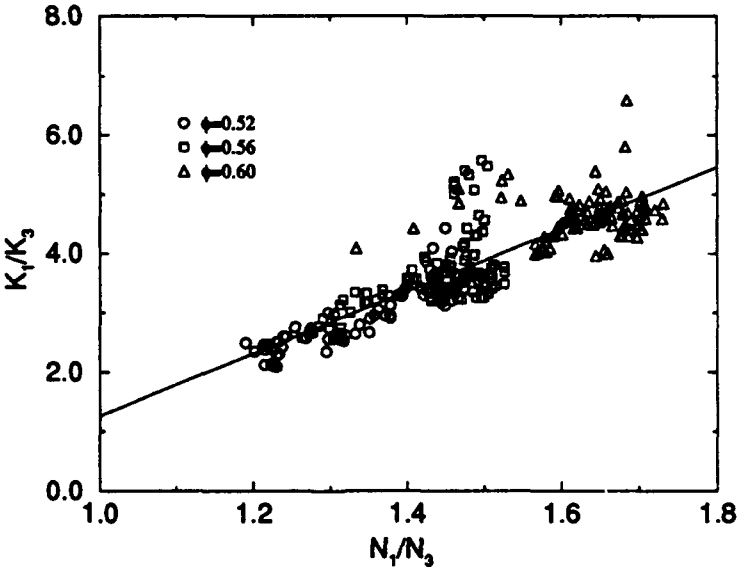


Figure 6.6: Relations between principal value ratios of conductivity and fabric (slope=5.23).

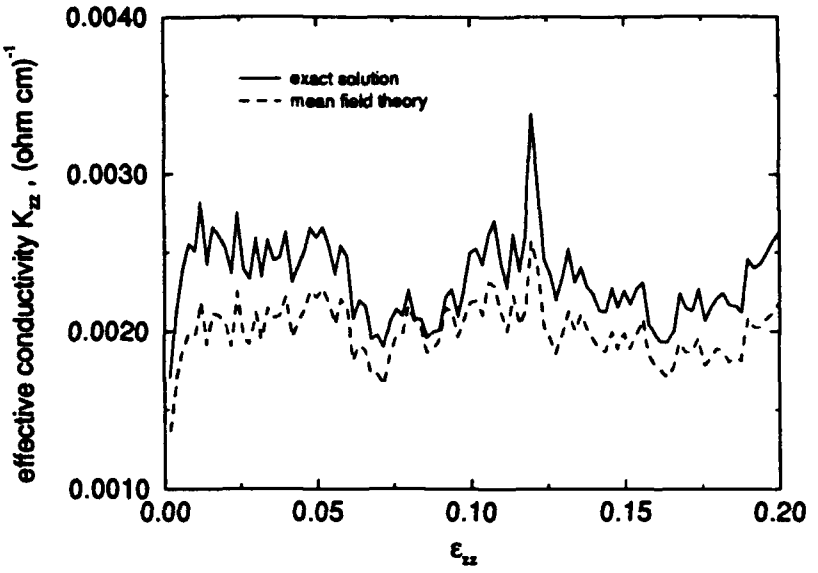


Figure 6.7: Comparison between the exact solution and mean field theory in predicting the effective conductivity of idealized granular assemblages ($\phi = 0.60$).

6.2 Experimental Verifications

Three triaxial compression tests have been carried out, one with 'clean' balls and two with 'dirty' balls, at approximate $\phi = 0.60$ initial densities. Here, clean balls refer to those, whose protective oil film was just removed and cleaned by Aceton solvent, and are expected to have no much contamination film on the ball surface (or low contact resistance, however, the normal load-contact resistance relation was not experimentally determined immediately after cleaning process unfortunately), whereas the dirty balls refer to those, which have been exposed to normal laboratory environment for approximately four months after being cleaned with Aceton, thereby, possess thicker insulating contamination film (or high contact resistance), and for which the load-contact resistance relation (Fig. 4.4) and interparticle friction coefficient have been measured. The experimental results with clean balls are given here solely for the comparison against the experimental results with dirty balls. It is not intended to compare the experimental results with clean balls to the results of the numerical

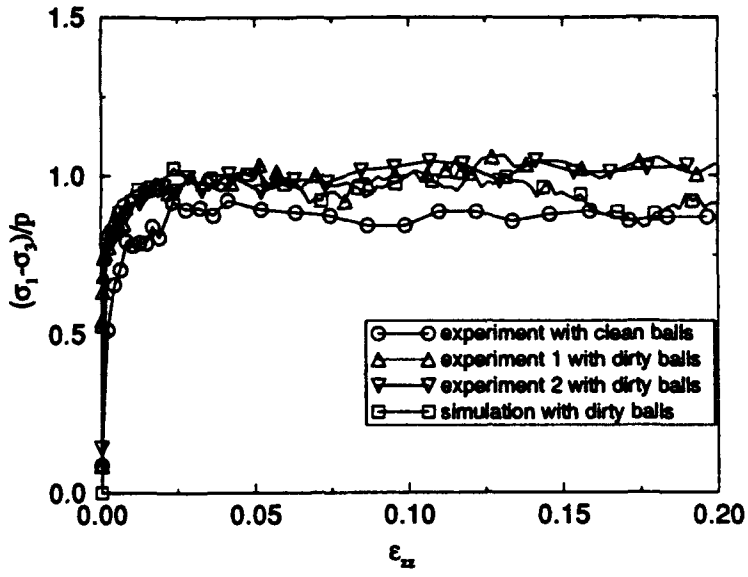


Figure 6.8: Comparison of shear strength between numerical simulation and experiments with dirty balls ($\phi = 0.60$).

simulations, since no numerical results for clean balls are available.

Plots Fig. 6.8 and 6.9 indicate that the results of numerical simulation and physical experiments with dirty steel balls are in qualitative agreement, and that the simulation is capable of predicting the mechanical and scalar transport properties of granular assemblages. Comparison of the experimental results between clean and dirty balls in Fig. 6.9 also reveals that the individual contact resistance can drastically affect the conductivity of the medium.

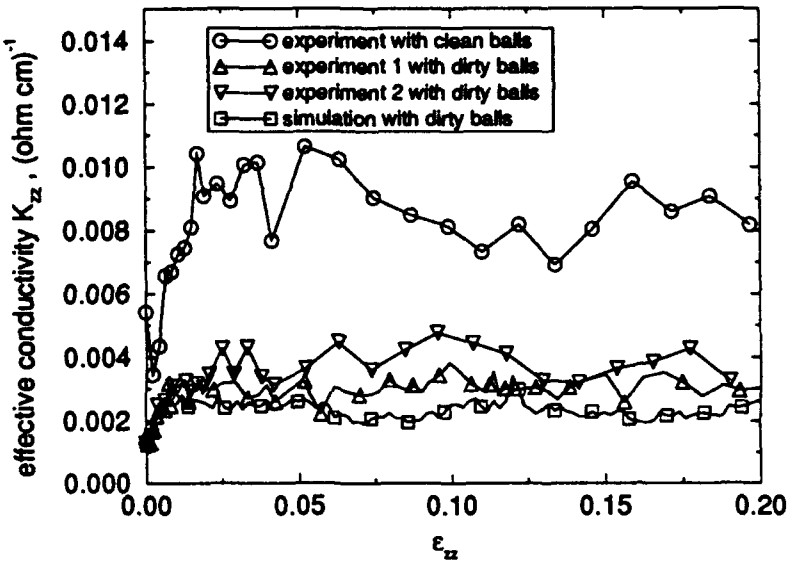


Figure 6.9: Comparison of electrical conductivity between numerical simulation and experiments with dirty balls ($\phi = 0.60$).

Chapter 7

Conclusions and Recommendations

We have developed a new version of a quasi-static simulation of the mechanics and conductivity of sphere assemblages by introducing several new numerical techniques, including a relaxation method which is shown to be a useful tool to overcome a singularity in the quasi-linear system of equilibrium equations. The computer code is versatile enough to allows one to simulate any deformation history and to study both mechanical and scalar transport properties of idealized granular assemblages simultaneously.

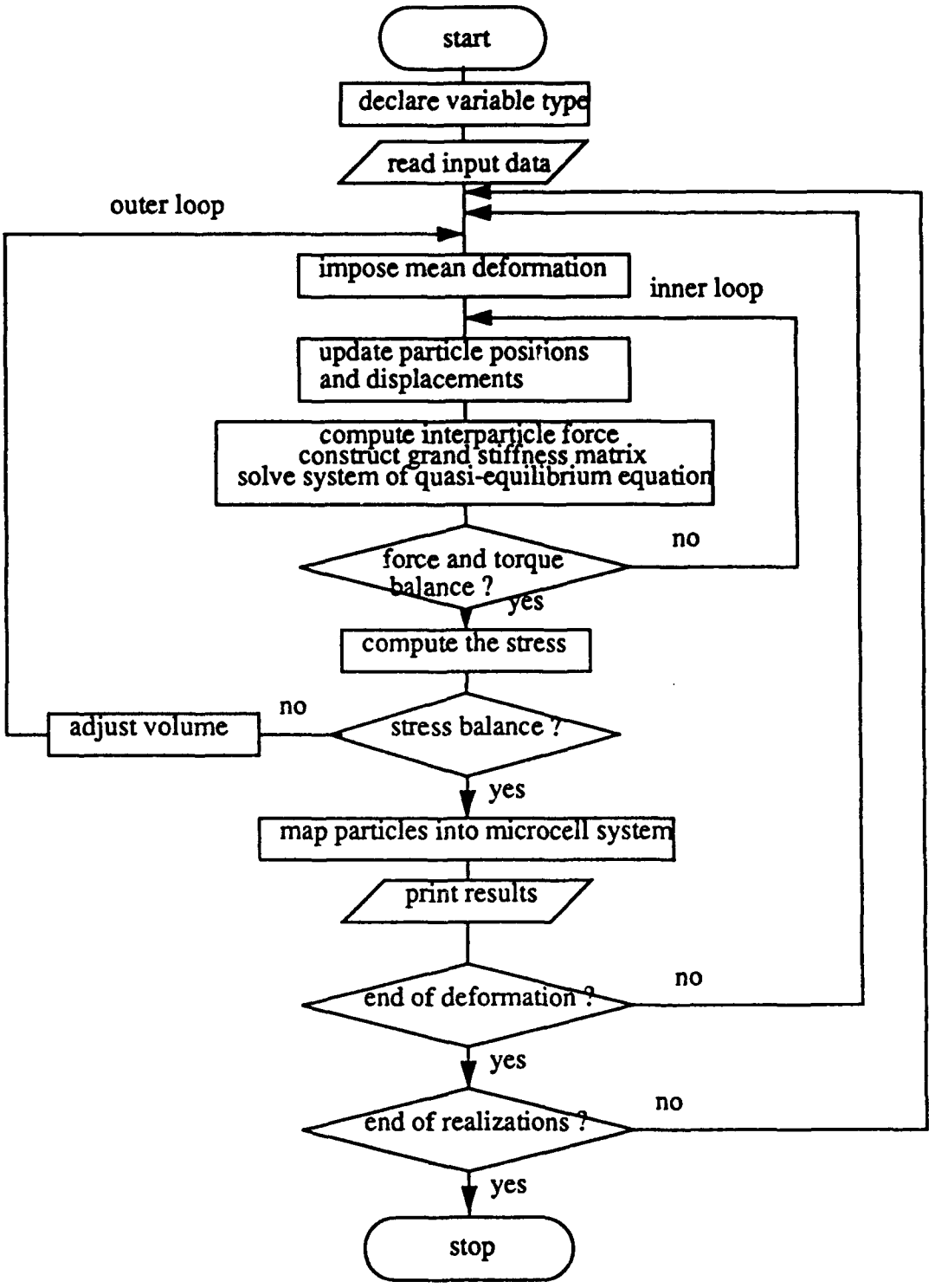
The results of the present investigation show that: (1) interparticle friction has great influence on Reynolds dilatancy for random dense-packed mono- as well as poly-disperse granular assemblages, a result contrary to Reynolds' original hypothesis; (2) the use of linear contact mechanics is justified near the ideal rigid-particle limit; (3) scalar transport properties such as electrical conductivity can be employed as a good indicator of the stress anisotropy and microstructural particle-contact topology, (4) the comparison between numerical results and experimental findings reveals that the numerical model is able to qualitatively predict the mechanical as well as scalar transport properties of idealized granular assemblages; (5) the contact resistance between steel balls deviates greatly from the theoretical prediction and depends strongly

on the normal load.

Although the present numerical algorithm can simulate sphere assemblages consisting of multiple-sized particles, an extension to include ellipsoid particles is called for in order to study the particle shape effects on the mechanical behavior as well as scalar transport properties of granular assemblages. To examine the Hertzian-contact resistance theory, one should find new type of particles with better surface smoothness and no surface resistance film. One of the promising materials is ion-exchange beads, provided that they are not too soft.

Appendix A

Flow Chart for the Numerical Algorithm


start
declare variable type
read input data
outer loop
impose mean deformation
inner loop
update particle positions
and displacements
compute interparticle force
construct grand stiffness matrix
solve system of quasi-equilibrium equation
force and torque
balance ?
no
yes
compute the stress
adjust volume
no
stress balance ?
yes
map particles into microcell system
print results
end of deformation ?
no
yes
end of realizations ?
no
yes
stop

Appendix B

Computer Code

```
      program UNIV3D

c---Variable definition:
c    dm--number of dimensions
c    npt--number of particles
c    idim--number of equations in quasi-linear system of equation
c    nl--number of microcells in X, Y and Z directions
c    mxnc--maximum number of microcells
c    mxc--maximum number of contacts a particle could possibly have
c    nnb--the number of adjecent microcells around a microcell
c    r--radii of particles
c    x,y,z--position coordinates of particles at prior step
c    xm,ym,zm--position coordinates of particles given by mean deformation
c    xt,yt,zt--position coordinates of particles given by "mean + fluctuation"
c    ux,uy,uz,wx,wy,wz--fluctuation displacement
c    uxm,uym,uzm,wxm,wym,wzm--mean displacement
c    uxt,uyt,uzt,wxt,wyt,wzt--total displacement
c    a--grand stiffness matrix
c    b--vector of unbalanced force
c    ak,bk--temparory 6 by 6 matrix
c    xn--correction to the fluctuating displacement
c    dxn--increment of xn
c    res--residual of the system of quasi-linear equations
c    fn0--normal force between particles at prior step
c    fn1--normal force between particles at current step
c    ft0--tangential force between particles at prior step
c    ft1--tangential force between particles at current step
c    tft--total tangential force between particles at current step
c    dft--increment of tangential force between particles at current step
c    tfor--total force between particles at current step
c    sfx,sfy,sfz--total force on each particle
c    smx,smy,smz--total moment on each particle
c    nij--fabric tensor
c    di--distance between adjacent particles
c    nx,ny,nz--contact normal
c    s--stress tensor
c    strn0,strn--deviatoric strain rate tensor
```

```
c   sstrn,psstrn--total strain
c   shear--shear strain increment
c   cond--conductance matrix
c   brx,bry,brz--net unbalanced current due to mean potential gradient
c   cur--interparticle current
c   voltf--fluctuating potential
c   c_eff--effective conductivity tensor
c   ih--number of contacts for a particle
c   adjc--adjacency matrix of microcell
c   kxx,kyy,kzz--for the periodicity of simulation cell
c   map0--microcells occupied by npt particles at prior step
c   map--microcells occupied by npt particles at current step
c   mappart0--particles occupying the microcells at prior step
c   mappart--particles occupying the microcells at current step
c   kphi,ktheta--contact normal distribution
c   ctpx,ctpy,ctpz--coordinates of contact points
c   thkn,thkt--scaled magnitude of normal and tangential force
c   msh0,msh1--random sequences of microcells
c   khlf1,khlf2--first and second half of a random sequence

c---Definition of input variables

c   iprepk--job option(0 for packing, 1 for repacking, 2 for relaxing,
c          3 for deformation)
c   irmax--number of realizations
c   istp--number of deformation steps
c   outmax--maximum number of outer loops
c   inmax--maximum number of inner loops
c   nrecord1,nrecord2--number of data points to be saved
c   shearm--shear strain increment
c   sigma0--controlling pressure
c   epsa--a small number
c   epso--allowance for the pressure balance
c   epsi--allowance for the force balance
c   den_i--initial density of loose packing
c   den0--desired packing density
c   iseed--seed for random number generator
c   strn0--deviatoric strain rate tensor
c   r1,r2,r3--particle radii
c   ckn,ckt--normal and tangential contact stiffness
c   cs--interparticle friction
c   orf--overrelaxation factor
c   irel--maximum number of iterations in relaxation method
c   nshuf--number of shuffling
c   lflip--number of flip in shuffling
c   nrif--number of riffle in shuffling
c   itrplmax--maximum number of trials in placing a particle in a microcell
c            without overlaping adjecent particles
c   devm1 to dv8--controling parameters for packing
c   i_cumu--an integer number
c   itrialm--maximum number of trials of packing to desired density
c   e_ym--elastic modulus
```

```
c   pois_r--Poisson's ratio
c   barkn--scaling factor of normal contact stiffness
c   barR--scaling factor for radii
c   grad--potential gradient
c   conk,conb--two parameters in load-resistance relation
c   isigm--pressure control parameter


      implicit real (a-h,o-z)
      implicit integer (i-n)
      integer dm

c      parameter(dm=3,npt=48,idim=6*npt,nl=8,mxnc=nl**dm,mxc=12)
c      parameter(dm=3,npt=48,idim=6*npt,nl=10,mxnc=nl**dm,mxc=12)
      parameter(dm=3,npt=100,idim=6*npt,nl=9,mxnc=nl**dm,mxc=12)
      parameter(nnb=342,pi=3.1415927,zero=0.0,izero=0,one=1.0,val=1000.)
      real x(npt),y(npt),z(npt),r(npt)
      real xm(npt),ym(npt),zm(npt),xt(npt),yt(npt),zt(npt)
      real ux(npt),uy(npt),uz(npt),uxm(npt),uym(npt),uzm(npt)
      real wx(npt),wy(npt),wz(npt),wxm(npt),wym(npt),wzm(npt)
      real uxt(npt),uyt(npt),uzt(npt),wxt(npt),wyt(npt),wzt(npt)
      real a(idim,idim),b(idim),ak(6,6),bk(6,6),xn(idim)

      real fn0(npt,npt),fn1(npt,npt),tft(npt,npt)
      real ft0(npt,npt,dm),ft1(npt,npt,dm),dft(dm)
      real sfx(npt),sfy(npt),sfz(npt),smx(npt),smy(npt),smz(npt)
      real tfor(npt,npt),d1(npt,npt),dxn(idim)
      real nx(npt,npt),ny(npt,npt),nz(npt,npt),nij(dm,dm)

      real res(idim),s(dm,dm)
      real ctpx(npt,mxc),ctpy(npt,mxc),ctpz(npt,mxc)
      real strn0(3,3),strn(3,3),sstrn(3,3),psstrn(3,3),starin(3,3)

      real cond(npt,npt),brx(npt),bry(npt),brz(npt)
      real cur(npt,npt),voltf(npt),c_eff(3,3)
      real cxn(npt),cres(npt),cdxn(npt)
      integer ih(npt),kxx(mxnc,nnb),kyy(mxnc,nnb),kzz(mxnc,nnb)
      integer adjc(mxnc,nnb),mcont(npt,npt),map(npt),map0(npt)
      integer mappart(mxnc),mappart0(mxnc)
      integer stest,outmax,highstrn,kphi(12),ktheta(24)
      integer thkn(npt,mxc),thkt(npt,mxc)
      integer msh0(mxnc),msh1(mxnc),khlf1(mxnc),khlf2(mxnc)

      open(unit=10,file='3dat',status='old')

      open(unit=30,file='condu',status='unknown')

c      open(unit=49,file='inner',status='unknown')
c      open(unit=50,file='outer',status='unknown')
c      open(unit=51,file='displ',status='unknown')
      open(unit=53,file='slope',status='unknown')
      open(unit=54,file='phi',status='unknown')
```

```
      open(unit=55,file='nbond',status='unknown')
      open(unit=56,file='theta',status='unknown')

      open(unit=61,file='predat0',status='unknown')
      open(unit=62,file='coord0',status='unknown')
      open(unit=71,file='predat1',status='unknown')
      open(unit=72,file='coord1',status='unknown')

      open(unit=68,file='fnav0',status='unknown')
      open(unit=69,file='fnav',status='unknown')
      open(unit=70,file='fn',status='unknown')

      open(unit=80,file='coord',status='unknown')
      open(unit=81,file='contc',status='unknown')
      open(unit=82,file='contp',status='unknown')
      open(unit=83,file='thick',status='unknown')
      open(unit=92,file='force',status='unknown')
c      open(unit=98,file='balan',status='unknown')
      open(unit=99,file='problem',status='unknown')

c---reading initial data and parameters

      read(10,*)iprepk,irmax,istp,outmax,inmax,nrecord1,nrecord2
      read(10,*)shearm,sigma0,epsa,epso,epsi,den_i,den0,iseed
      read(10,*)strn0(1,1),strn0(1,2),strn0(1,3)
      read(10,*)strn0(2,1),strn0(2,2),strn0(2,3)
      read(10,*)strn0(3,1),strn0(3,2),strn0(3,3)
      read(10,*)r1,r2,r3,ckn,ckt,cs,orf,irel
      read(10,*)nshuf,lflip,nrif,itrplmax,devm1,devm2,devm3,af1,af2
      read(10,*)den02,den01,den1,den2,den3,den4,den5
      read(10,*)dv0,dv1,dv2,dv3,dv4,dv5,dv6,dv7,dv8
      read(10,*)i_cumu,itrialm,e_ym,pois_r,barkn,barR,grad,amda
      read(10,*)conk,conb,isigm

      write(*,*)'isigm',isigm

      if(nrecord1.ne.0)then
         idist1=istp/nrecord1
      else
         idist1=10000
      end if
      if(nrecord2.ne.0)then
         idist2=istp/nrecord2
      else
         idist2=10000
      end if
      if(istp.le.nrecord1)idist1=1
      if(istp.le.nrecord2)idist2=1
      monop=1
      if(r1.ne.r2.or.r2.ne.r3)monop=2

      if(iprepk.le.2)then
```

```
         epsi=0.0
         cs=0.0
         call rinit2(3,3,strn0,0.0)
         if(den0.le.0.6)shearm=0.0
      end if

c---carrying out "irmax" realizations

      do 599 ir=1,irmax

         itrial=1

 199  icumu=0

c---generating initial loose system for packing

         if(iprepk.eq.0)then

            istold=0
            ev=0.0

            n20=nint(real(npt)/(1.0+(r2/r1)**dm+(r2/r3)**dm))
            n10=nint(real(n20)*((r2/r1)**dm))
            n30=npt-n10-n20
            write(*,*)n10,n20,n30
            p1=real(n10)/real(npt)
            p2=real(n10+n20)/real(npt)
            solidv=4.0*pi*(n10*r1**dm+n20*r2**dm+n30*r3**dm)/3.0

            iseed=(iseed+int(100000*ran(iseed)))*2+1
            ratioxz=1.0
            ratioyz=1.0
            strmax=shearm
            delz0=2.0*r1/sqrt((ratioxz+tan(strmax))**2+ratioyz**2+1.0)
            delx0=delz0*ratioxz
            dely0=delz0*ratioyz
            nm=1+exp(log(solidv/den_i)/dm)/delz0
 101     write(*,*)'initial nm:',nm

 103        if(nm.gt.nl)then
               write(*,*)'nm>nl,not enough microcell.','nm,nl=',nm,nl
               write(99,*)'nm>nl,not enough microcell.','nm,nl=',nm,nl
               stop
            end if
            dx0=delx0*nm
            dy0=dely0*nm
            dz0=delz0*nm
            ncell=nm**dm
            ijs0=1+2.0*r3/exp(log(solidv/(ncell*den0))/dm)
            if(real(ijs0).ge.real(nm)/2.0)then
               nm=nm+1
               go to 101
```

```
            end if

            call adj3(mxnc,ijs0,nm,nnb,adjc,kxx,kyy,kzz)
            call shuffle(ncell,nshuf,lflip,nrif,iseed,msh0,
     +     msh1,khlf1,khlf2)

            call vec_$iinit(mappart0,mxnc,izero)

            n1=0
            n2=0
            n3=0
            kap=0
            ncount=0

            do 100 np=1,npt

               if(monop.eq.1)then
                  r(np)=r2
               else
           write(*,*)'please include r_assign subroutine'
                  call r_assign(n1,n2,n3,n10,n20,n30,r1,r2,r3,
     +            p1,p2,iseed,r(np))
               end if

 83            kap=kap+1
               if(kap.gt.ncell)then
c                 write(*,*)'ir=',ir,'  nm=',nm,'  np=',np,'  kap=',kap
                  nm=nm+1
                  go to 103
               end if

               k1=msh0(kap)

               km=k1/(nm*nm)
               if(km*nm*nm.ne.k1)km=km+1
               jm=(k1-(km-1)*nm*nm)/nm
               if(jm*nm.ne.(k1-(km-1)*nm*nm))jm=jm+1
               im=k1-(km-1)*nm*nm-(jm-1)*nm
               ncount=ncount+1

               hx=(im-1)*delx0
               hy=(jm-1)*dely0
               hz=(km-1)*delz0

               itrpl=0
 84            itrpl=itrpl+1
               x(np)=hx+ran(iseed)*delx0
               y(np)=hy+ran(iseed)*dely0
               z(np)=hz+ran(iseed)*delz0

               do 80 j=1,nnb
                  kn=mappart0(adjc(k1,j))
```

```
                  if(kn.eq.0)go to 80
                  k2=map0(kn)
                  xkn=x(kn)-kxx(k1,j)*dx0
                  ykn=y(kn)-kyy(k1,j)*dy0
                  zkn=z(kn)-kzz(k1,j)*dz0
                  d02=(x(np)-xkn)**2+(y(np)-ykn)**2+(z(np)-zkn)**2
                  if(d02.ge.(r(np)+r(kn))*(r(np)+r(kn)))go to 80
                  if(itrpl.lt.itrplmax)then
                     go to 84
                  else
                     go to 83
                  end if
 80            continue
               map0(np)=k1
               mappart0(k1)=np

 100        continue

c-------------------------------------------------
            call rinit2(npt,npt,fn0,zero)
            call rinit3(npt,npt,dm,ft0,zero)
c            write(*,*)'ncount=',ncount
            density=solidv/(dx0*dy0*dz0)

            write(62,*)ir,ist,dx0,dy0,dz0,nm,ijs0
            do 79 i=1,npt
               write(62,*)i,r(i),x(i),y(i),z(i)
 79         continue

c---starting from a packed system to repack, to relax the stress,
c   to do simple shear etc.

         else

            write(*,*)'start from packed system'
            call rinit2(npt,npt,fn0,zero)
            call rinit3(npt,npt,dm,ft0,zero)

            read(61,*)irt,n10,n20,n30,p1,p2,density,solidv,sigma
            read(61,*)iu,istold,sstrn(1,1),sstrn(1,2),sstrn(1,3),
     +           sstrn(2,1),sstrn(2,2),sstrn(2,3),
     +           sstrn(3,1),sstrn(3,2),sstrn(3,3),ev,delev_last,iside0

            do 149 i=1,npt
               read(61,*)it,ntem
               if(ntem.eq.0)go to 149
               do 147 j=1,ntem
                  read(61,*)jla,fn0(i,jla),ft0(i,jla,1),
     +          ft0(i,jla,2),ft0(i,jla,3)
 147           continue
 149        continue
```

```
            read(62,*)irt,istt,dx0,dy0,dz0,nmt,ijs0t
            do 176 i=1,npt
               read(62,*)it,r(i),x(i),y(i),z(i)
 176        continue

            if(iprepk.eq.1)then
               densp=density
               delden=(den0-densp)/10.0
               nden=1
            end if

            ratioxz=dx0/dz0
            ratioyz=dy0/dz0
            if(iprepk.eq.3)then
            strmax=max(abs(strn0(1,2)),abs(strn0(1,3)),abs(strn0(2,3)),
     +   abs(strn0(2,1)),abs(strn0(3,1)),abs(strn0(3,2)))*(istold+istp)
            else
               strmax=0.4
            end if
            delz0=2.0*r1/sqrt((ratioxz+tan(strmax))**2+ratioyz**2+1.0)
            nm=dz0/delz0+1

 177        if(nm.gt.nl)then
               write(*,*)'nm>nl,not enough microcell.','nm,nl=',nm,nl
               write(99,*)'nm>nl,not enough microcell.','nm,nl=',nm,nl
               stop
            end if

            write(*,*)'computed nm',nm,'  nl',nl
c           stop

            nm=nl
            delz0=dz0/nm
            delx0=delz0*ratioxz
            dely0=delz0*ratioyz
            ncell=nm**dm
            ijs0=1+2.0*r3/exp(log(solidv/(ncell*den0))/dm)
            if(real(ijs0).ge.real(nm)/2.0)then
               nm=nm+1
               go to 177
            end if
            write(*,*)'nm=',nm,',  ijs0=',ijs0,',  ncell=',ncell

            call adj3(mxnc,ijs0,nm,nnb,adjc,kxx,kyy,kzz)
            call mapping(npt,nm,mxnc,sstrn,delx0,dely0,delz0,
     +    x,y,z,map0,mappart0,iouts)

         end if

         kflag=0
c         write(98,*)ir
```

```
      call rinit2(3,3,sstrn,0.0)
      call rinit2(3,3,psstrn,0.0)

      ilarg=1
      ismal=0
c     if(iprepk.eq.2)istp=1

c---a number of cyclic shearings for packing or steps for simple shear

      do 598 ist=istold+1,istold+istp

         icheck=0

         iter3=0
         lowstrn=0
         highstrn=0
         nostop=0

692      if(iter3.ne.3.and.nostop.eq.0)then
            icheck=icheck+1
            call rmatx_copy(3,3,strn0,strn)
         else
            if(lowstrn.lt.2)then
               icheck=icheck+1
               do 690 i=1,3
          do 690 j=1,3
          strn(i,j)=strn(i,j)/2.0
690            continue
               lowstrn=lowstrn+1
       iter3=0
               if(nostop.eq.1)nostop=0

            else
       if(highstrn.eq.0)then
                  icheck=icheck+1
                  do 691 i=1,3
     do 691 j=1,3
                   strn(i,j)=strn0(i,j)*2.0
691                 continue
       else
                  icheck=icheck+1
                  do 693 i=1,3
     do 693 j=1,3
                   strn(i,j)=strn(i,j)*2.0
693                 continue
       end if

               if(nostop.eq.1)nostop=0
       iter3=0

               highstrn=highstrn+1
               if(highstrn.gt.2)then
```

```
      write(99,*)'sorry, I cannot find a satisfactory delev'
      write(99,*)'to balance the stress, stop! highstrn'
      stop
           end if

        end if

      end if

     if(iprepk.le.1)then
        if(den0.le.0.6.or.(den0.gt.0.6.and.density.gt.den0-0.002
     +       .and.abs(psstrn(3,1)+psstrn(1,2)).lt.1.0e-8))then
           strn(3,1)=0.0
           strn(1,2)=0.0
        else

           ismal=ismal+1
           if(mod(ilarg,2).ne.0)then

              if(ismal.eq.1)strn(3,1)=shearm
              if(ismal.eq.2)strn(3,1)=-shearm
              if(ismal.eq.3)strn(3,1)=-shearm
              if(ismal.eq.4)strn(3,1)=shearm
              strn(1,2)=0.0
              sstrn(1,2)=0.0
           else
              if(ismal.eq.1)strn(1,2)=shearm
              if(ismal.eq.2)strn(1,2)=-shearm
              if(ismal.eq.3)strn(1,2)=-shearm
              if(ismal.eq.4)strn(1,2)=shearm
              strn(3,1)=0.0
              sstrn(3,1)=0.0
           end if
           if(mod(ismal,4).eq.0)then
              ilarg=ilarg+1
              ismal=0
           end if

        end if
     end if

      if(icheck.gt.12)then
     write(99,*)'program terminated due to unconverged outloop'
     write(6,*)'program terminated due to unconverged outloop'
        stop
      end if

      strn_c=0.0
      sstrn_c=0.0
      do 687 i=1,3
         do 687 j=1,3
         sstrn(i,j)=psstrn(i,j)+strn(i,j)
```

```
                if(abs(strn(i,j)).gt.abs(strn_c))strn_c=strn(i,j)
                if(abs(sstrn(i,j)).gt.abs(sstrn_c))sstrn_c=sstrn(i,j)
 687          continue

c             write(99,*)ir,ist,iterout,strn_c
              iterout=0
c              write(50,*)'ir,ist,iterout,stest,test,error,delev,ev,flu/m,
c     +dens'
              write(*,*)'ir,ist,iterout,stest,test,error,delev,ev,flu/m,
     +dens, isigm'
              write(*,*)' '
c             write(98,*)' '


c---choosing the increment of contraction for packing

              if(iprepk.le.1)then
                 if(iprepk.eq.0.and.ist.le.1)then
            delev0=-dv0
                 else if(density.ge.den0)then
            delev0=0.0
            icumu=icumu+1
                 else
                    if(density.lt.den02)then
               delev0=-dv1
                 else if(density.lt.den01.and.density.ge.den02)then
               delev0=-dv2
                 else if(density.lt.den1.and.density.ge.den01)then
               delev0=-dv3
                 else if(density.lt.den2.and.density.ge.den1)then
               delev0=-dv4
                 else if(density.lt.den3.and.density.ge.den2)then
               delev0=-dv5
                 else if(density.lt.den4.and.density.ge.den3)then
               delev0=-dv6
                 else if(density.lt.den5.and.density.ge.den4)then
               delev0=-dv7
            else
               delev0=-dv8
            end if
                 end if
                 delev=delev0

c---do packing, force balancing, and computing stress

       call delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
     + iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
     + inmax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
     + delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
     + r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
     + wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
     + fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
```

```
     + a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
     + xn,dxn,res,orf,numr,mappart0,mappart,
     + s,test,sigma,stest,error,errormin,delevmin,kflag,amda)
               go to 604
            end if

c---algorithm to achieve the desirsd isotropic stress

            iter2=1
            iter3=0
            inega=1
            iwent=0

            dev1=zero
            delev=dev1
            if(iprepk.eq.2)then
c              iterout=iterout+1
c              test=sigma-sigma0
c              f_dev1=test
c              if(test.lt.0.0)then
c                 stest=-1
c              else
c                 stest=1
c              end if
c              error=test/sigma0
            else
               iterout=iterout+1

      call delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
     + iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
     + inmax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
     + delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
     + r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
     + wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
     + fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
     + a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
     + xn,dxn,res,orf,numr,mappart0,mappart,
     + s,test,sigma,stest,error,errormin,delevmin,kflag,amda)
         if(abs(error).lt.epso)go to 604

               f_dev1=test
            end if
            iside=stest
            test0=test
            error0=error

 592        igo942=0
               if(iside*inega.eq.-1)then
                  if(ist.le.1.or.iwent.eq.1)then
                     devm_1=devm1
                  else if(iter3.ge.1)then
                     devm_1=delevmin-0.0005*iter3
```

```
                else
                   if(iside0.eq.0)then
                      devm_1=devm1
                   else if(iside0.eq.-1)then
                      devm_1=af1*delev_last
                   else
                      devm_1=-af1*delev_last*4.0
                   end if
                end if
                delev=devm_1
                iterout=iterout+1

 call delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
+ iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
+ inmax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
+ delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
+ r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
+ wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
+ fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
+ a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
+ xn,dxn,res,orf,numr,mappart0,mappart,
+ s,test,sigma,stest,error,errormin,delevmin,kflag,amda)
    if(abs(error).lt.epso)go to 604

                fevm1=test
                if(ist.le.1.or.iwent.eq.1)then
                   devm_2=devm2
          iwent=0
                else if(iter3.ge.1)then
                   devm_2=delevmin-0.0001*iter3
                else
                   if(iside0.eq.0)then
                      devm_2=devm2
                   else if(iside0.eq.-1)then
                      devm_2=af2*delev_last
                   else
                      devm_2=-af2*delev_last*4.0
                   end if
                end if
                delev=devm_2
                iterout=iterout+1

 call delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
+ iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
+ inmax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
+ delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
+ r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
+ wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
+ fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
+ a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
+ xn,dxn,res,orf,numr,mappart0,mappart,
+ s,test,sigma,stest,error,errormin,delevmin,kflag,amda)
```

```
        if(abs(error).lt.epso)go to 604

                fevm2=test
         go to 791
              else
942             if(((iside0.le.0.or.error0.lt.1.0).and.iter3.eq.0).
    +   or.igo942.eq.1)then
                    devm_2=0.0
                    fevm2=test0
                    devm_1=devm3
                    delev=devm_1
941                 iterout=iterout+1

     call delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
    + iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
    + inmax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
    + delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
    + r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
    + wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
    + fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
    + a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
    + xn,dxn,res,orf,numr,mappart0,mappart,
    + s,test,sigma,stest,error,errormin,delevmin,kflag,amda)
        if(abs(error).lt.epso)go to 604

                    fevm1=test
                    if(fevm1.gt.fevm2)then
                       go to 791
                    else
                       devm_1=10.0*devm_1
       delev=devm_1
                       go to 941
                    end if
                 else
c--------
                    if(iter3.eq.0)then
                       devm_1=delev_last/af1
                    else
                       devm_1=delevmin/(af1*iter3)
                    end if
                    delev=devm_1
                    iterout=iterout+1

     call delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
    + iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
    + inmax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
    + delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
    + r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
    + wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
    + fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
    + a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
    + xn,dxn,res,orf,numr,mappart0,mappart,
```

```
     + s,test,sigma,stest,error,errormin,delevmin,kflag,amda)
         if(abs(error).lt.epso)go to 604

                     fevm1=test

                     if(iter3.eq.0)then
                        devm_2=delev_last/af2
                     else
                        devm_2=delevmin/(af2*iter3)
                     end if
                     delev=devm_2
                     iterout=iterout+1

          call delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
     + iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
     + imax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
     + delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
     + r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
     + wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
     + fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
     + a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
     + xn,dxn,res,orf,numr,mappart0,mappart,
     + s,test,sigma,stest,error,errormin,delevmin,kflag,amda)
         if(abs(error).lt.epso)go to 604

                     fevm2=test

                     if(fevm1.gt.fevm2.and.fevm1.gt.0.0)then
                        go to 791
                     else
                        if(test0.gt.fevm2)then
                           fevm1=test0
                           devm_1=0.0
                           go to 791
                        else if(test0.gt.fevm1)then
                           fevm2=fevm1
                           devm_2=devm_1
                           fevm1=test0
                           devm_1=0.0
                           go to 791
                        else
               igo942=1
                           go to 942
                        end if
                     end if
c--------
                  end if
               end if

 791           if(fevm1*fevm2.lt.0.0)then
            if(fevm1.gt.0.0)then
              devl=devm_2
```

```
      devh=devm_1
      f_devl=fevm2
      f_devh=fevm1
    else
      devl=devm_1
      devh=devm_2
      f_devl=fevm1
      f_devh=fevm2
    end if
    go to 940
        end if
        slpk=(fevm1-fevm2)/(devm_1-devm_2)
        if(abs(slpk).lt.0.02)then
    if(iside*inega.eq.-1)then
      devm_1=devm_1-0.002
      delev=devm_1
              iterout=iterout+1
      if(iter2.eq.1.and.iterout.gt.30)then
  nostop=1
  go to 692
        end if

 call delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
+ iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
+ inmax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
+ delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
+ r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
+ wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
+ fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
+ a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
+ xn,dxn,res,orf,numr,mappart0,mappart,
+ s,test,sigma,stest,error,errormin,delevmin,kflag,amda)
    if(abs(error).lt.epso)go to 604

        fevm1=test
      go to 791
    else
      if(devm_1.le.0.0)then
              devm_1=2.0*devm_1
      else
              devm_1=0.5*devm_1
      end if
            delev=devm_1
            iterout=iterout+1

      if(iter2.eq.1.and.iterout.gt.30)then
  nostop=1
  go to 692
            end if

 call delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
+ iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
```

```
+ inmax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
+ delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
+ r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
+ wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
+ fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
+ a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
+ xn,dxn,res,orf,numr,mappart0,mappart,
+ s,test,sigma,stest,error,errormin,delevmin,kflag,amda)
    if(abs(error).lt.epso)go to 604

                fevm1=test
       go to 791
     end if

          else

          ilow=0
          ihigh=0
     devm_3=devm_1-fevm1/slpk
     if(iside*inega.eq.-1)then
        if(devm_3.lt.0.0)then
           delev=devm_3
        else
           delev=-0.00005
        end if
        devm_3=delev
     else
        if(devm_3.gt.0.001)devm_3=0.0005
        delev=devm_3
     end if
             iterout=iterout+1

        if(iter2.eq.1.and.iterout.gt.30)then
   nostop=1
   go to 692
                end if

 call delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
+ iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
+ inmax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
+ delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
+ r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
+ wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
+ fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
+ a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
+ xn,dxn,res,orf,numr,mappart0,mappart,
+ s,test,sigma,stest,error,errormin,delevmin,kflag,amda)
    if(abs(error).lt.epso)go to 604

     if(test.le.0.0)then
                ilow=1
                devl=delev
```

```
                f_devl=test
                slpk=(fevm2-test)/(devm_2-delev)
                gevm_3=delev
                gevm3=test
                if(abs(slpk).lt.0.02)then
                   slpk=(fevm1-test)/(devm_1-delev)
                end if
                delev=delev-test/slpk
                iterout=iterout+1

          if(iter2.eq.1.and.iterout.gt.30)then
     nostop=1
     go to 692
                end if

 call delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
+ iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
+ inmax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
+ delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
+ r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
+ wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
+ fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
+ a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
+ xn,dxn,res,orf,numr,mappart0,mappart,
+ s,test,sigma,stest,error,errormin,delevmin,kflag,amda)
     if(abs(error).lt.epso)go to 604
     go to 929

      else
                ihigh=1
                devh=delev
                f_devh=test
                slpk=(fevm2-test)/(devm_2-delev)
                gevm_3=delev
                gevm3=test
                delev=delev-test/slpk
                iterout=iterout+1

          if(iter2.eq.1.and.iterout.gt.30)then
     nostop=1
     go to 692
                end if

 call delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
+ iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
+ inmax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
+ delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
+ r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
+ wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
+ fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
+ a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
+ xn,dxn,res,orf,numr,mappart0,mappart,
```

```
     + s,test,sigma,stest,error,errormin,delevmin,kflag,amda)
         if(abs(error).lt.epso)go to 604
         go to 929

          end if
929       if(test.le.0.0)then
                    ilow=1
                    devl=delev
                    f_devl=test
                    if(ilow*ihigh.eq.1)go to 940
                    slpk=(gevm3-test)/(gevm_3-delev)
                    gevm_3=delev
                    gevm3=test
                    if(abs(slpk).lt.0.02)then
                       slpk=(fevm2-test)/(devm_2-delev)
                    end if
                    delev=delev-test/slpk
                    iterout=iterout+1

              if(iter2.eq.1.and.iterout.gt.30)then
            nostop=1
            go to 692
                    end if

          call delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
     + iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
     + inmax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
     + delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
     + r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
     + wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
     + fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
     + a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
     + xn,dxn,res,orf,numr,mappart0,mappart,
     + s,test,sigma,stest,error,errormin,delevmin,kflag,amda)
         if(abs(error).lt.epso)go to 604
         go to 929

          else
                    ihigh=1
                    devh=delev
                    f_devh=test
                    if(ilow*ihigh.eq.1)go to 940
                    slpk=(gevm3-test)/(gevm_3-delev)
                    gevm_3=delev
                    gevm3=test
                    if(abs(slpk).lt.0.02)then
                       slpk=(fevm2-test)/(devm_2-delev)
                    end if
                    delev=delev-test/slpk
                    iterout=iterout+1

              if(iter2.eq.1.and.iterout.gt.30)then
```

```
      nostop=1
      go to 692
                  end if

       call delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
     + iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
     + inmax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
     + delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
     + r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
     + wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
     + fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
     + a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
     + xn,dxn,res,orf,numr,mappart0,mappart,
     + s,test,sigma,stest,error,errormin,delevmin,kflag,amda)
         if(abs(error).lt.epso)go to 604
         go to 929

          end if

               end if

940          ddev=devh-devl
             errormin=100.0
             delevmin=0.0
             iter2=2
             do 924 iterout=1,outmax
                rtflsp=devl+ddev*f_devl/(f_devl-f_devh)
                if(abs(rtflsp-delevt).lt.0.0000001)then
                   delev=rtflsp*1.50
                else
                   delev=rtflsp
                end if
                delevt=delev

       call delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
     + iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
     + inmax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
     + delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
     + r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
     + wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
     + fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
     + a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
     + xn,dxn,res,orf,numr,mappart0,mappart,
     + s,test,sigma,stest,error,errormin,delevmin,kflag,amda)
         if(abs(error).lt.epso)go to 604

              f_f=test
              if(f_f.lt.0.0)then
           devl=delev
           f_devl=f_f
              else
           devh=delev
```

```
          f_devh=f_f
               end if
               ddev=devh-devl
 924        continue

            iter3=iter3+1
            iterout=1
            iter2=1
            if(iter3.lt.2)then
               go to 592
            else if(iter3.eq.2.and.iside.gt.0)then
               inega=-1
               go to 592
            else if(iter3.eq.2.and.iside.lt.0)then
               iwent=1
               go to 592
            else if(iter3.eq.3.and.lowstrn.lt.2)then
               go to 692
            else if(iter3.eq.3.and.highstrn.lt.2)then
               go to 692
            else
               write(99,*)'sorry, I cannot find a satisfactory delev'
               write(99,*)'to balance the stress, stop! end of outloop'
               stop
            end if

c---updating the position, force, mapping etc. for the next step
c---computing and storing

 604        call vec_$copy(xt,x,npt)
            call vec_$copy(yt,y,npt)
            call vec_$copy(zt,z,npt)
            call vec_$icopy(map,map0,npt)
            call vec_$icopy(mappart,mappart0,mxnc)
            call rmatx_copy(npt,npt,fn1,fn0)
            call rmatx3_copy(npt,npt,dm,ft1,ft0)

            delx0=delx
            dely0=dely
            delz0=delz
            dx=delx0*nm
            dy=dely0*nm
            dz=delz0*nm

            ev=ev+delev
            delev_last=delev
            if(delev.gt.0.0)iside0=1
            if(abs(delev).lt.1.0e-10)iside0=0
            if(delev.lt.0.0)iside0=-1

            if(iprepk.ge.3)then
               if(abs(strn_c).gt.1.0e-8)slop=delev/strn_c
```

```
            end if

            call vec_$iinit(kphi,12,0)
            call vec_$iinit(ktheta,24,0)

c            if((iprepk.gt.1.and.ist.eq.(idist2*
c     +         (ist/idist2))).or.kflag.eq.1)then
c              call rinit2(npt,mxc,ctpx,zero)
c              call rinit2(npt,mxc,ctpy,zero)
c              call rinit2(npt,mxc,ctpz,zero)
c              call init2(npt,mxc,thkn,izero)
c              call init2(npt,mxc,thkt,izero)
c            end if

c            do 607 i=1,npt-1
c               do 608 j=i+1,npt
c         d1(j,i)=d1(i,j)
c 608          continue
c 607       continue

            call copyhalf(npt,d1)

            call rinit2(dm,dm,nij,zero)

            call rinit2(npt,npt,cond,zero)
            call vec_$init(brx,npt,zero)
            call vec_$init(bry,npt,zero)
            call vec_$init(brz,npt,zero)

            call init2(npt,npt,mcont,0)
            call rinit2(npt,npt,tfor,zero)
            call rinit2(npt,npt,tft,zero)

            condmax=0.0
            condmin=100.0
            condmax2=0.0
            condmin2=100.0
            avr_fn=0.0
            navr_fn=0

            avr_cond2=0.0
            navr_cond2=0

            do 221 np=1,npt
               ih(np)=0
               do 222 j_kn=1,nnb

          kn=mappart(adjc(map(np),j_kn))
          if(kn.eq.0)go to 222
                 if(fn1(np,kn).ge.-epsa)go to 222

          nij(1,1)=nij(1,1)+nx(np,kn)*nx(np,kn)
```

```
                nij(1,2)=nij(1,2)+nx(np,kn)*ny(np,kn)
                nij(1,3)=nij(1,3)+nx(np,kn)*nz(np,kn)
                nij(2,1)=nij(2,1)+ny(np,kn)*nx(np,kn)
                nij(2,2)=nij(2,2)+ny(np,kn)*ny(np,kn)
                nij(2,3)=nij(2,3)+ny(np,kn)*nz(np,kn)
                nij(3,1)=nij(3,1)+nz(np,kn)*nx(np,kn)
                nij(3,2)=nij(3,2)+nz(np,kn)*ny(np,kn)
                nij(3,3)=nij(3,3)+nz(np,kn)*nz(np,kn)

                      mcont(np,kn)=1
                      tfor(np,kn)=sqrt(fn1(np,kn)**2+ft1(np,kn,1)**2+
     +          ft1(np,kn,2)**2+ft1(np,kn,3)**2)
                if(abs(cs).gt.1.0e-5)then
                         tft(np,kn)=sqrt(ft1(np,kn,1)**2+
     +           ft1(np,kn,2)**2+ft1(np,kn,3)**2)
                end if

                         kx=kxx(map(np),j_kn)
                         ky=kyy(map(np),j_kn)
                         kz=kzz(map(np),j_kn)

                         xl=kx*nm*delx+ky*tan(sstrn(2,1))*nm*dely+
     +      kz*tan(sstrn(3,1))*nm*delz
                         yl=ky*nm*dely+kx*tan(sstrn(1,2))*nm*delx+
     +      kz*tan(sstrn(3,2))*nm*delz
                         zl=kz*nm*delz+kx*tan(sstrn(1,3))*nm*delx+
     +                ky*tan(sstrn(2,3))*nm*dely
                         xkn=xt(kn)-xl
                         ykn=yt(kn)-yl
                         zkn=zt(kn)-zl

                       call degree_phi(nz(np,kn),kphi)
               call degree_theta(nx(np,kn),ny(np,kn),ktheta)

c                      if((iprepk.gt.1.and.ist.eq.(idist2*
c     +                    (ist/idist2))).or.kflag.eq.1)then
c                       ih(np)=ih(np)+1
c                       ctpx(np,ih(np))=(r(kn)*xt(np)+r(np)*xkn)/d1(np,kn)
c                       ctpy(np,ih(np))=(r(kn)*yt(np)+r(np)*ykn)/d1(np,kn)
c                       ctpz(np,ih(np))=(r(kn)*zt(np)+r(np)*zkn)/d1(np,kn)
c
c                      end if

                    if(iprepk.eq.3.or.(iprepk.le.2.and.kflag.eq.1))then

                freal=abs(fn1(np,kn)*barR*barkn)
                cond(np,kn)=conb*exp(conk*log(freal))

                avr_cond2=avr_cond2+cond(np,kn)
                navr_cond2=navr_cond2+1

                avr_fn=avr_fn+fn1(np,kn)
```

```
          navr_fn=navr_fn+1

c         if(kx.ne.0.or.ky.ne.0)cond(np,kn)=0.0

          if(cond(np,kn).gt.condmax)condmax=cond(np,kn)
          if(cond(np,kn).lt.condmin.and.cond(np,kn).gt.1.0e-20)
     +        condmin=cond(np,kn)
          brx(np)=brx(np)-cond(np,kn)*(xkn-x(np))*barR*grad
          bry(np)=bry(np)-cond(np,kn)*(ykn-y(np))*barR*grad
          brz(np)=brz(np)-cond(np,kn)*(zkn-z(np))*barR*grad

              end if

 222          continue

 221        continue

c-----------------------------------------
          if(iprepk.eq.3.or.(iprepk.le.2.and.kflag.eq.1))then
            do 224 i=1,npt
               ptm=0.0
               do 223 j=1,npt
          ptm=ptm+cond(i,j)
 223           continue
               cond(i,i)=-ptm
 224         continue

           call conduct(npt,mxnc,nnb,dx,dy,dz,sstrn,x,y,z,r,
     +      nx,ny,nz,cond,brx,bry,brz,cur,voltf,c_eff,
     +      cxn,cres,cdxn,map,mappart,adjc,kxx,kyy,kzz,grad,barR)

            write(30,*)ist,sstrn_c,condmax,condmin
            write(30,*)'  ',c_eff(1,1),c_eff(1,2),c_eff(1,3)
            write(30,*)'  ',c_eff(2,1),c_eff(2,2),c_eff(2,3)
            write(30,*)'  ',c_eff(3,1),c_eff(3,2),c_eff(3,3)
            write(30,*)'  '

c------ computing conductivity by mean field theory

            avr_fn=avr_fn/real(navr_fn)

            avr_freal=abs(avr_fn*barR*barkn)
            avr_cond=conb*exp(conk*log(avr_freal))

            cmc=2.0*r2*r2*avr_cond/(dx*dy*dz*barR)

            avr_cond2=avr_cond2/real(navr_cond2)
            cmc2=2.0*r2*r2*avr_cond2/(dx*dy*dz*barR)
       write(30,*)'  ',avr_fn,avr_cond,avr_cond2,cmc2*nij(3,3)
c           write(30,*)'  ',avr_fn,avr_freal,avr_cond,cmc
            write(30,*)'  ',cmc*nij(1,1),cmc*nij(1,2),cmc*nij(1,3)
            write(30,*)'  ',cmc*nij(2,1),cmc*nij(2,2),cmc*nij(2,3)
```

```
            write(30,*)'  ',cmc*nij(3,1),cmc*nij(3,2),cmc*nij(3,3)
            write(30,*)'  '

          end if
c-------------------------------------

            do 323 i=1,dm
               do 324 j=1,dm
                  nij(i,j)=nij(i,j)/(2.0*nbon)
 324            continue
 323         continue

c--------------------------

            fmax=0.0
            fmin=10.0
            fnmax=0.0
            fnmin=10.0
            ftmax=0.0
            ftmin=10.0

            do 258 i=1,npt-1
               do 259 j=i+1,npt

                  if(fn1(i,j).gt.-epsa)go to 259
                  if(tfor(i,j).gt.fmax)then
                     fmax=tfor(i,j)
                     imax=i
                     jmax=j
                  end if
                  if(tfor(i,j).lt.fmin)then
                     fmin=tfor(i,j)
                     imin=i
                     jmin=j
                  end if

                  if(abs(fn0(i,j)).gt.fnmax)fnmax=abs(fn0(i,j))
                  if(abs(fn0(i,j)).lt.fnmin)fnmin=abs(fn0(i,j))
                  if(tft(i,j).gt.ftmax)ftmax=tft(i,j)
                  if(tft(i,j).lt.ftmin)ftmin=tft(i,j)
 259            continue
 258         continue

             if(ir.eq.1.or.kflag.eq.1)then
                write(70,*)ir,ist,nbon,fnmax,fnmin,-avr_fn
                do 358 i=1,npt-1
                   do 359 j=i+1,npt
                      if(fn1(i,j).gt.-epsa)go to 359
              write(70,*)fn1(i,j)
 359                continue
 358             continue
             end if
```

```
            if(ist.eq.idist1*(ist/idist1).or.kflag.eq.1)then

               write(54,*)ir,ist
               do 855 k=1,12
                  fraction=kphi(k)/(2.0*nbon+1.0e-10)
                  write(54,863)k*15.0,fraction
 863              format(2x,i6,f10.3)
 855           continue

               write(56,*)ir,ist
               do 856 k=1,24
                  fraction=ktheta(k)/(2.0*nbon+1.0e-10)
                  write(56,863)k*15.0,fraction
 856           continue

c               write(51,*)ir,ist
c               do 589 i=1,npt
c                  write(51,57)uxt(i),uyt(i),uzt(i)
c         write(51,57)wxt(i),wyt(i),wzt(i)
c 57               format(2x,3f15.6)
c 589           continue

               if(fmin.ge.epsa.and.fmin.ne.10.0)then
                  write(92,*)ist,' fn,  ft'
                 write(92,*)'   Smallest force:',imin,jmin,'fmin= ',fmin
                 write(92,*)'   Largest force:',imax,jmax,'fmax= ',fmax
                 write(92,*)'   Smallest normal force:',' fnmin= ',fnmin
                 write(92,*)'   Largest normal force:','  fnmax= ',fnmax

c                  do 265 i=1,npt
c                     write(92,*)i
c                     do 267 j=1,npt
c                        if(fn0(i,j).lt.-epsa)then
c               write(92,266)j,fn0(i,j),ft0(i,j,1),ft0(i,j,2),ft0(i,j,3)
c 266                      format(5x,i4,4(f15.11,1x))
c                        end if
c 267                 continue
c 265              continue

               end if
            end if

            write(55,854)ir,ist,sstrn_c,nbon,kslp,kact
 854        format(2x,2i5,f12.6,3i5)

            write(68,*)-sstrn_c,-fnaverage
            write(69,*)-sstrn_c,-fnav_cur

            npart=0
            do 189 k=1,npt
               if(map(k).ge.1.and.map(k).le.ncell)then
```

```
                  npart=npart+1
               end if
 189        continue

            write(53,851)ir,ist,npart,sstrn_c,delev,
     +         density,slop,ev
 851        format(1x,3i5,5f10.5)
            write(53,852)s(1,1),s(1,2),s(1,3)
            write(53,852)s(2,1),s(2,2),s(2,3)
            write(53,852)s(3,1),s(3,2),s(3,3)
            write(53,*)' '

            write(53,852)nij(1,1),nij(1,2),nij(1,3)
            write(53,852)nij(2,1),nij(2,2),nij(2,3)
            write(53,852)nij(3,1),nij(3,2),nij(3,3)
 852        format(5x,3(1x,e15.7))

c            if((iprepk.gt.1.and.ist.eq.(idist2*
c     +         (ist/idist2))).or.kflag.eq.1)then
c
c               write(80,*)ir,ist,delx*nm,dely*nm,delz*nm,
c     +    sstrn(1,2),sstrn(1,3),sstrn(2,3)
c     +    sstrn(2,1),sstrn(3,1),sstrn(3,2)
c                do 489 i=1,npt
c                   write(80,339)i,r(i),x(i),y(i),z(i)
c 339              format(i5,f6.2,3(1x,f10.5))
c 489           continue
c               write(80,*)kflag
c
c               fndif=(fnmax-fnmin)/4.0
c               ftdif=(ftmax-ftmin)/4.0
c
c               call init2(npt,mxc,thkn,izero)
c               call init2(npt,mxc,thkt,izero)
c
c               do 702 i=1,npt
c                  jq=0
c                  do 701 j=1,npt
c
c                     if(fn0(i,j).gt.-epsa)go to 701
c                     jq=jq+1
c            if(abs(fndif).le.epsa)then
c       thkn(i,jq)=1
c            else
c                  thkn(i,jq)=int((abs(fn0(i,j))-fnmin)/fndif)+1
c            end if
c
c                     if(abs(tft(i,j)).gt.epsa.and.
c     +       abs(ftdif).le.epsa)then
c                  thkt(i,jq)=1
c                     else if(abs(tft(i,j)).gt.epsa.and.
c     +       abs(ftdif).gt.epsa)then
```

```
c                  thkt(i,jq)=int((abs(tft(i,j))-ftmin)/ftdif)+1
c                     else
c                        thkt(i,jq)=0
c                     end if
c
c 701                continue
c 702             continue
c
c                 write(81,*)ist
c                 do 86 i=1,npt
c                    write(81,87)i,ih(i)
c 87                 format(2x,2i5)
c 86              continue
c
c                 write(82,*)ist
c                 write(83,*)ist,fnmax,fnmin,ftmax,ftmin
c                 do 97 i=1,npt
c                    write(82,88)(ctpx(i,j),j=1,10)
c                    write(82,88)(ctpy(i,j),j=1,10)
c                    write(82,88)(ctpz(i,j),j=1,10)
c                    write(83,89)(thkn(i,j),j=1,10)
c                    write(83,89)(thkt(i,j),j=1,10)
c 88                 format(10(1x,f6.3))
c 89                 format(10(1x,i2))
c 97              continue
c
c              end if

c              if(iprepk.gt.2)then
c                  close(71,status='delete')
c                  close(72,status='delete')
c                  open(unit=71,file='predat1',status='unknown')
c                  open(unit=72,file='coord1',status='unknown')
c                 iwr=1
c              end if

               do 98 i=1,3
                  do 98 j=1,3
                  psstrn(i,j)=sstrn(i,j)
                  starin(i,j)=sstrn(i,j)
                  if(iprepk.le.2)starin(i,j)=0.0
 98            continue

               istol=ist
               evv=ev
               delev_la=delev_last
               if(iprepk.le.2)then
                  istol=0
                  evv=0.0
                  delev_la=0.0
               end if
```

```
          if(iprepk.eq.1.and.abs(sstrn(3,1)+sstrn(1,2)).lt.1.0e-8.and.
     +       density.ge.(densp+nden*delden).and.density.lt.
     +       (densp+(nden+1)*delden))then
             nden=nden+1
             iwr=1
          end if

c         if(kflag.eq.1.or.(ist.eq.istold+istp).or.iwr.eq.1.or.
c     +       (icumu.ge.i_cumu.and.iprepk.eq.1))then
c            iwr=0
c             write(71,*)ir,n10,n20,n30,p1,p2,dy,solidv,sigma
c             write(71,*)ist,istol,starin(1,1),starin(1,2),starin(1,3),
c     + starin(2,1),starin(2,2),starin(2,3),
c     + starin(3,1),starin(3,2),starin(3,3),evv,delev_la,iside0

c
c            do 192 i=1,npt
c               ntem=0
c               do 191 k=1,npt
c                  ntem=mcont(i,k)+ntem
c 191           continue
c
c               write(71,*)i,ntem
c               do 193 j=1,npt
c                  if(fn0(i,j).lt.0.0)then
c               write(71,*)j,fn0(i,j),ft0(i,j,1),ft0(i,j,2),ft0(i,j,3)
c                  end if
c 193           continue
c 192        continue
c
c            write(72,*)ir,ist,dx,dy,dz,nm,ijs0
c            do 196 i=1,npt
c               write(72,*)i,r(i),x(i),y(i),z(i)
c 196        continue
c         end if

c            if(kflag.eq.1)go to 599
c           if(icumu.ge.i_cumu.and.iprepk.eq.0)then
c              if(itrial.ge.itrialm)then
c         write(99,*)'maximum number of trials to pack'
c         write(99,*)'to desired density is reached, stop!'
c         stop
c              else
c                 itrial=itrial+1
c                 go to 199
c              end if
c           end if
c           if(icumu.ge.i_cumu.and.iprepk.eq.1)go to 599
 598     continue

            write(68,*)'&'
            write(69,*)'&'
```

```
c        if(iprepk.eq.2)then
c           write(*,*)'ir=',ir,',   density=',density
c        end if

 599  continue

      stop
      end

c-----------------------------------------------------------------------

c---subroutine to do packing, and force balancing

      subroutine delev_stress(dm,monop,npt,idim,nm,iprepk,ir,ist,den0,
     + iterout,inner,iter2,iter3,idist1,zero,val,ckn,ckt,cs,
     + inmax,epsi,epso,epsa,irel,sigma0,isigm,fnaverage,fnav_cur,
     + delev,ev,sstrn,strn,delx0,dely0,delz0,delx,dely,delz,mxnc,nnb,
     + r,x,y,z,xm,ym,zm,xt,yt,zt,ux,uy,uz,uxm,uym,uzm,uxt,uyt,uzt,
     + wx,wy,wz,wxm,wym,wzm,wxt,wyt,wzt,solidv,volume,density,
     + fn0,fn1,ft0,ft1,dft,d1,sfx,sfy,sfz,smx,smy,smz,itrial,
     + a,b,ak,bk,adjc,kxx,kyy,kzz,nx,ny,nz,map0,map,nbon,kslp,kact,
     + xn,dxn,res,orf,numr,mappart0,mappart,
     + s,test,sigma,stest,error,errormin,delevmin,kflag,amda)

      implicit real (a-h,o-z)
      implicit integer (i-n)

      integer dm
      real x(npt),y(npt),z(npt),r(npt)
      real xm(npt),ym(npt),zm(npt),xt(npt),yt(npt),zt(npt)
      real ux(npt),uy(npt),uz(npt),uxm(npt),uym(npt),uzm(npt)
      real wx(npt),wy(npt),wz(npt),wxm(npt),wym(npt),wzm(npt)
      real uxt(npt),uyt(npt),uzt(npt),wxt(npt),wyt(npt),wzt(npt)
      real a(idim,idim),b(idim),ak(6,6),bk(6,6),xn(idim)

      real fn0(npt,npt),fn1(npt,npt)
      real ft0(npt,npt,dm),ft1(npt,npt,dm),dft(dm)
      real sfx(npt),sfy(npt),sfz(npt),smx(npt),smy(npt),smz(npt)
      real mx0,my0,mz0,d1(npt,npt),dxn(idim)
      real nx(npt,npt),ny(npt,npt),nz(npt,npt),nxx,nyy,nzz
      real res(idim),s(dm,dm),sstrn(3,3),strn(3,3)
c      real res(idim),s(dm,dm),sstrn(6),strn(6)

      integer kxx(mxnc,nnb),kyy(mxnc,nnb),kzz(mxnc,nnb)
      integer adjc(mxnc,nnb),map(npt),map0(npt),stest
      integer mappart(mxnc),mappart0(mxnc)

      ncell=nm**dm
      cknt=ckn-ckt

c        write(*,*)'step=',ist,' out=',iterout,' iter2=',iter2,
```

```
c     +' iter3=',iter3
c        write(6,*)'  inner, bx, by, bz, sumb, ratb, nb, nbs, stress'

c            if(ist.eq.idist1*(ist/idist1))then
c              write(49,*)'step=',ist,' out=',iterout,' iter2=',iter2,
c     +        ' iter3=',iter3
c              write(49,*)'  inner, bx, by, bz, sumb, ratb, nb,
c     + nbs, stress'
c            end if

c---move particles according to mean displacement

            factorx=strn(1,1)
            factory=strn(2,2)
            factorz=strn(3,3)

            factor=delev/dm

            do 714 i=1,npt
              uxm(i)=(factorx+factor)*x(i)+strn(2,1)*y(i)+strn(3,1)*z(i)
              uym(i)=(factory+factor)*y(i)+strn(1,2)*x(i)+strn(3,2)*z(i)
              uzm(i)=(factorz+factor)*z(i)+strn(1,3)*x(i)+strn(2,3)*y(i)
               wxm(i)=strn(2,3)-strn(3,2)
               wym(i)=strn(3,1)-strn(1,3)
               wzm(i)=strn(1,2)-strn(2,1)
               xm(i)=x(i)+uxm(i)
               ym(i)=y(i)+uym(i)
               zm(i)=z(i)+uzm(i)
               ux(i)=0.0
               uy(i)=0.0
               uz(i)=0.0
               wx(i)=0.0
               wy(i)=0.0
               wz(i)=0.0
 714         continue

            delx=delx0*(1.0+factorx+factor)
            dely=dely0*(1.0+factory+factor)
            delz=delz0*(1.0+factorz+factor)
            density=solidv/(ncell*delx*dely*delz)

            if(iprepk.le.1.and.density.lt.0.5)th
               inmaxo=inmax/10
            else
               inmaxo=inmax
            end if

c---inner loop to balance the force

            iway=1

            inner=0
```

```
333       inner=inner+1

          call vec_$add_vector(xm,ux,npt,xt)
          call vec_$add_vector(ym,uy,npt,yt)
          call vec_$add_vector(zm,uz,npt,zt)

          call vec_$add_vector(uxm,ux,npt,uxt)
          call vec_$add_vector(uym,uy,npt,uyt)
          call vec_$add_vector(uzm,uz,npt,uzt)

          call vec_$add_vector(wxm,wx,npt,wxt)
          call vec_$add_vector(wym,wy,npt,wyt)
          call vec_$add_vector(wzm,wz,npt,wzt)

          if(inner.eq.1)then
             call vec_$icopy(map0,map,npt)
             call vec_$icopy(mappart0,mappart,mxnc)
          else
             call mapping(npt,nm,mxnc,sstrn,delx,dely,delz,
     +   xt,yt,zt,map,mappart,iouts)
             if(iouts.ge.1)then
           write(99,*)'particles fly out of the system!!'
           write(99,*)ir,ist,iterout,inner,' iouts=',iouts
                 write(99,*)ist,delx*nm,dely*nm,delz*nm,
     +        sstrn(1,1),sstrn(1,2),sstrn(1,3),
     +        sstrn(2,1),sstrn(2,2),sstrn(2,3),
     +        sstrn(3,1),sstrn(3,2),sstrn(3,3)
          do 334 i=1,npt
             write(99,*)i,r(i),xt(i),yt(i),zt(i)
334       continue
             end if
          end if

          abfx=0.0
          abfy=0.0
          abfz=0.0
          abmx=0.0
          abmy=0.0
          abmz=0.0
          kslp=0
          kact=0

          call rinit2(npt,npt,fn1,zero)
          call rinit3(npt,npt,dm,ft1,zero)
          call rinit2(npt,npt,d1,val)
          call vec_$init(sfx,npt,zero)
          call vec_$init(sfy,npt,zero)
          call vec_$init(sfz,npt,zero)
          call vec_$init(smx,npt,zero)
          call vec_$init(smy,npt,zero)
          call vec_$init(smz,npt,zero)
```

```
            if(inner.le.inmaxo)then
               call rinit2(idim,idim,a,zero)
               call vec_$init(b,idim,zero)
            end if

            nbon=0
            if(iway.eq.1)fnaverage=0.0
               fnav_cur=0.0
c---computing grand stiffness and unbalanced force vector
c---evaluating contact forces between particle

            do 21 np=1,npt
               knp=6*(np-1)
               do 22 j_kn=1,nnb

          kn=mappart(adjc(map(np),j_kn))
          if(kn.eq.0)go to 22
          if(d1(np,kn).lt.100.0.or.d1(kn,np).lt.100.0)go to 22

                  kx=kxx(map(np),j_kn)
                  ky=kyy(map(np),j_kn)
                  kz=kzz(map(np),j_kn)

                     xl=kx*nm*delx+ky*tan(sstrn(2,1))*nm*dely+
     +  kz*tan(sstrn(3,1))*nm*delz
                     yl=ky*nm*dely+kx*tan(sstrn(1,2))*nm*delx+
     +  kz*tan(sstrn(3,2))*nm*delz
                     zl=kz*nm*delz+kx*tan(sstrn(1,3))*nm*delx+
     +          ky*tan(sstrn(2,3))*nm*dely
                  xkn=xt(kn)-xl
                  ykn=yt(kn)-yl
                  zkn=zt(kn)-zl

                  d12=(xkn-xt(np))**2+(ykn-yt(np))**2+
     +       (zkn-zt(np))**2
                  if(d12.gt.(r(np)+r(kn)+0.001)*(r(np)+r(kn)+0.001))go to 22
                  d1(np,kn)=sqrt(d12)

                  ovlap=d1(np,kn)-r(np)-r(kn)

          if(iway.eq.2)then

             ckn=exp(amda*log(abs(ovlap/fnaverage)))
             ckt=0.8*ckn
          end if

                  fn=ckn*ovlap
                  if(fn.gt.-epsa)then
             go to 22
                  end if

                  fn1(np,kn)=fn
```

```
      if(iway.eq.1)fnaverage=fnaverage+fn
           fnav_cur=fnav_cur+fn

              nx(np,kn)=(xkn-xt(np))/d1(np,kn)
              ny(np,kn)=(ykn-yt(np))/d1(np,kn)
              nz(np,kn)=(zkn-zt(np))/d1(np,kn)
              nxx=nx(np,kn)
              nyy=ny(np,kn)
              nzz=nz(np,kn)
      rax=r(np)*nxx
      ray=r(np)*nyy
      raz=r(np)*nzz

      uxnp=uxt(np)
      uynp=uyt(np)
      uznp=uzt(np)
      wxnp=r(np)*wxt(np)
      wynp=r(np)*wyt(np)
      wznp=r(np)*wzt(np)

      uxkn=uxt(kn)-(factorx+factor)*xl-strn(2,1)*yl-strn(3,1)*zl
      uykn=uyt(kn)-(factory+factor)*yl-strn(1,2)*xl-strn(3,2)*zl
      uzkn=uzt(kn)-(factorz+factor)*zl-strn(1,3)*xl-strn(2,3)*yl
      wxkn=r(kn)*wxt(kn)
      wykn=r(kn)*wyt(kn)
      wzkn=r(kn)*wzt(kn)

      uxr=uxkn-uxnp
      uyr=uykn-uynp
      uzr=uzkn-uznp
      wxr=wxnp+wxkn
      wyr=wynp+wykn
      wzr=wznp+wzkn

      nbon=nbon+1
              if(abs(cs).lt.1.0e-8)then
         ft1(np,kn,1)=0.0
         ft1(np,kn,2)=0.0
         ft1(np,kn,3)=0.0
         kslp=kslp+1
      else

         dft(1)=ckt*((1.-nxx**2)*uxr-nxx*nyy*uyr-nxx*nzz*uzr
     +        -nzz*wyr+nyy*wzr)
                   dft(2)=ckt*(-nyy*nxx*uxr+(1.-nyy**2)*uyr-nyy*nzz*uzr
     +        +nzz*wxr-nxx*wzr)
        dft(3)=ckt*(-nzz*nxx*uxr-nzz*nyy*uyr+(1.-nzz**2)*uzr
     +        -nyy*wxr+nxx*wyr)

        dft(1)=dft(1)+ft0(np,kn,1)*(1.-nxx**2)-ft0(np,kn,2)*
     +        nxx*nyy-ft0(np,kn,3)*nxx*nzz
```

```
      dft(2)=dft(2)-ft0(np,kn,1)*nyy*nxx+ft0(np,kn,2)*(1.-
     +    nyy*nyy)-ft0(np,kn,3)*nyy*nzz
      dft(3)=dft(3)-ft0(np,kn,1)*nzz*nxx-ft0(np,kn,2)*
     +    nzz*nyy+ft0(np,kn,3)*(1.-nzz*nzz)

       ft=sqrt(dft(1)**2+dft(2)**2+dft(3)**2)
       tx=dft(1)/ft
       ty=dft(2)/ft
       tz=dft(3)/ft

       if(abs(ft).lt.epsa)ft=0.0

            if(ft-cs*abs(fn).gt.0.0)then
               ft=cs*abs(fn)
               ft1(np,kn,1)=ft*tx
               ft1(np,kn,2)=ft*ty
               ft1(np,kn,3)=ft*tz
               kslp=kslp+1
            else
               ft1(np,kn,1)=dft(1)
               ft1(np,kn,2)=dft(2)
               ft1(np,kn,3)=dft(3)
               kact=kact+1
            end if

         end if

         fx=fn*nxx+ft1(np,kn,1)
         fy=fn*nyy+ft1(np,kn,2)
         fz=fn*nzz+ft1(np,kn,3)
         if(abs(cs).lt.1.0e-8)then
            mx0=0.0
            my0=0.0
            mz0=0.0
         else
            mx0=ray*ft1(np,kn,3)-raz*ft1(np,kn,2)
            my0=raz*ft1(np,kn,1)-rax*ft1(np,kn,3)
            mz0=rax*ft1(np,kn,2)-ray*ft1(np,kn,1)
         end if

         sfx(np)=sfx(np)+fx
         sfy(np)=sfy(np)+fy
         sfz(np)=sfz(np)+fz
         smx(np)=smx(np)+mx0
         smy(np)=smy(np)+my0
         smz(np)=smz(np)+mz0

         sfx(kn)=sfx(kn)-fx
         sfy(kn)=sfy(kn)-fy
         sfz(kn)=sfz(kn)-fz
         smx(kn)=smx(kn)+mx0*r(kn)/r(np)
         smy(kn)=smy(kn)+my0*r(kn)/r(np)
```

```
      smz(kn)=smz(kn)+mz0*r(kn)/r(np)

      abfx=abfx+abs(fx)
      abfy=abfy+abs(fy)
      abfz=abfz+abs(fz)
      abmx=abmx+abs(mx0)
      abmy=abmy+abs(my0)
      abmz=abmz+abs(mz0)

      if(inner.eq.(inmaxo+1))go to 22
      kkn=6*(kn-1)

      cknt=ckn-ckt

      ak(1,1)=cknt*nxx*nxx+ckt
      ak(1,2)=cknt*nxx*nyy
      ak(1,3)=cknt*nxx*nzz
      ak(1,4)=0.0
      ak(1,5)=-ckt*nzz
      ak(1,6)=ckt*nyy

      ak(2,1)=cknt*nyy*nxx
      ak(2,2)=(cknt*nyy*nyy+ckt)
      ak(2,3)=cknt*nyy*nzz
      ak(2,4)=ckt*nzz
      ak(2,5)=0.0
      ak(2,6)=-ckt*nxx

      ak(3,1)=cknt*nzz*nxx
      ak(3,2)=cknt*nzz*nyy
      ak(3,3)=(cknt*nzz*nzz+ckt)
      ak(3,4)=-ckt*nyy
      ak(3,5)=ckt*nxx
      ak(3,6)=0.0

    ak(4,1)=ckt*(-ray*nzz*nxx+raz*nyy*nxx)
    ak(4,2)=-ckt*(ray*nzz*nyy+raz*(1.0-nyy*nyy))
    ak(4,3)=ckt*(ray*(1.0-nzz*nzz)+raz*nyy*nzz)
    ak(4,4)=-ckt*(ray*nyy+raz*nzz)
    ak(4,5)=ckt*ray*nxx
    ak(4,6)=ckt*raz*nxx

    ak(5,1)=ckt*(raz*(1.0-nxx*nxx)+rax*nzz*nxx)
    ak(5,2)=ckt*(-raz*nxx*nyy+rax*nzz*nyy)
    ak(5,3)=-ckt*(raz*nxx*nzz+rax*(1.0-nzz*nzz))
    ak(5,4)=ckt*rax*nyy
    ak(5,5)=-ckt*(raz*nzz+rax*nxx)
    ak(5,6)=ckt*raz*nyy

    ak(6,1)=-ckt*(rax*nyy*nxx+ray*(1.0-nxx*nxx))
    ak(6,2)=ckt*(rax*(1.0-nyy*nyy)+ray*nxx*nyy)
    ak(6,3)=ckt*(-rax*nyy*nzz+ray*nxx*nzz)
```

```
ak(6,4)=ckt*rax*nzz
ak(6,5)=ckt*ray*nzz
ak(6,6)=-ckt*(rax*nxx+ray*nyy)

  knp1=knp+1
  knp2=knp+2
  knp3=knp+3
  knp4=knp+4
  knp5=knp+5
  knp6=knp+6
  kkn1=kkn+1
  kkn2=kkn+2
  kkn3=kkn+3
  kkn4=kkn+4
  kkn5=kkn+5
  kkn6=kkn+6

        a(knp1,knp1)=a(knp1,knp1)-ak(1,1)
        a(knp1,knp2)=a(knp1,knp2)-ak(1,2)
        a(knp1,knp3)=a(knp1,knp3)-ak(1,3)
        a(knp1,knp4)=a(knp1,knp4)+ak(1,4)*r(np)
        a(knp1,knp5)=a(knp1,knp5)+ak(1,5)*r(np)
        a(knp1,knp6)=a(knp1,knp6)+ak(1,6)*r(np)

        a(knp2,knp1)=a(knp2,knp1)-ak(2,1)
        a(knp2,knp2)=a(knp2,knp2)-ak(2,2)
        a(knp2,knp3)=a(knp2,knp3)-ak(2,3)
        a(knp2,knp4)=a(knp2,knp4)+ak(2,4)*r(np)
        a(knp2,knp5)=a(knp2,knp5)+ak(2,5)*r(np)
        a(knp2,knp6)=a(knp2,knp6)+ak(2,6)*r(np)

        a(knp3,knp1)=a(knp3,knp1)-ak(3,1)
        a(knp3,knp2)=a(knp3,knp2)-ak(3,2)
        a(knp3,knp3)=a(knp3,knp3)-ak(3,3)
        a(knp3,knp4)=a(knp3,knp4)+ak(3,4)*r(np)
        a(knp3,knp5)=a(knp3,knp5)+ak(3,5)*r(np)
        a(knp3,knp6)=a(knp3,knp6)+ak(3,6)*r(np)

        a(knp4,knp1)=a(knp4,knp1)-ak(4,1)
        a(knp4,knp2)=a(knp4,knp2)-ak(4,2)
        a(knp4,knp3)=a(knp4,knp3)-ak(4,3)
        a(knp4,knp4)=a(knp4,knp4)+ak(4,4)*r(np)
        a(knp4,knp5)=a(knp4,knp5)+ak(4,5)*r(np)
        a(knp4,knp6)=a(knp4,knp6)+ak(4,6)*r(np)

        a(knp5,knp1)=a(knp5,knp1)-ak(5,1)
        a(knp5,knp2)=a(knp5,knp2)-ak(5,2)
        a(knp5,knp3)=a(knp5,knp3)-ak(5,3)
        a(knp5,knp4)=a(knp5,knp4)+ak(5,4)*r(np)
        a(knp5,knp5)=a(knp5,knp5)+ak(5,5)*r(np)
        a(knp5,knp6)=a(knp5,knp6)+ak(5,6)*r(np)
```

```
a(knp6,knp1)=a(knp6,knp1)-ak(6,1)
a(knp6,knp2)=a(knp6,knp2)-ak(6,2)
a(knp6,knp3)=a(knp6,knp3)-ak(6,3)
a(knp6,knp4)=a(knp6,knp4)+ak(6,4)*r(np)
a(knp6,knp5)=a(knp6,knp5)+ak(6,5)*r(np)
a(knp6,knp6)=a(knp6,knp6)+ak(6,6)*r(np)

a(knp1,kkn1)=ak(1,1)
a(knp1,kkn2)=ak(1,2)
a(knp1,kkn3)=ak(1,3)
a(knp1,kkn4)=ak(1,4)*r(kn)
a(knp1,kkn5)=ak(1,5)*r(kn)
a(knp1,kkn6)=ak(1,6)*r(kn)

a(knp2,kkn1)=ak(2,1)
a(knp2,kkn2)=ak(2,2)
a(knp2,kkn3)=ak(2,3)
a(knp2,kkn4)=ak(2,4)*r(kn)
a(knp2,kkn5)=ak(2,5)*r(kn)
a(knp2,kkn6)=ak(2,6)*r(kn)

a(knp3,kkn1)=ak(3,1)
a(knp3,kkn2)=ak(3,2)
a(knp3,kkn3)=ak(3,3)
a(knp3,kkn4)=ak(3,4)*r(kn)
a(knp3,kkn5)=ak(3,5)*r(kn)
a(knp3,kkn6)=ak(3,6)*r(kn)

a(knp4,kkn1)=ak(4,1)
a(knp4,kkn2)=ak(4,2)
a(knp4,kkn3)=ak(4,3)
a(knp4,kkn4)=ak(4,4)*r(kn)
a(knp4,kkn5)=ak(4,5)*r(kn)
a(knp4,kkn6)=ak(4,6)*r(kn)

a(knp5,kkn1)=ak(5,1)
a(knp5,kkn2)=ak(5,2)
a(knp5,kkn3)=ak(5,3)
a(knp5,kkn4)=ak(5,4)*r(kn)
a(knp5,kkn5)=ak(5,5)*r(kn)
a(knp5,kkn6)=ak(5,6)*r(kn)

a(knp6,kkn1)=ak(6,1)
a(knp6,kkn2)=ak(6,2)
a(knp6,kkn3)=ak(6,3)
a(knp6,kkn4)=ak(6,4)*r(kn)
a(knp6,kkn5)=ak(6,5)*r(kn)
a(knp6,kkn6)=ak(6,6)*r(kn)

bk(1,1)=ak(1,1)
bk(1,2)=ak(1,2)
bk(1,3)=ak(1,3)
```

```
bk(1,4)=-ak(1,4)
bk(1,5)=-ak(1,5)
bk(1,6)=-ak(1,6)

bk(2,1)=ak(2,1)
bk(2,2)=ak(2,2)
bk(2,3)=ak(2,3)
bk(2,4)=-ak(2,4)
bk(2,5)=-ak(2,5)
bk(2,6)=-ak(2,6)

bk(3,1)=ak(3,1)
bk(3,2)=ak(3,2)
bk(3,3)=ak(3,3)
bk(3,4)=-ak(3,4)
bk(3,5)=-ak(3,5)
bk(3,6)=-ak(3,6)

bk(4,1)=-ak(4,1)*r(kn)/r(np)
bk(4,2)=-ak(4,2)*r(kn)/r(np)
bk(4,3)=-ak(4,3)*r(kn)/r(np)
bk(4,4)=ak(4,4)*r(kn)/r(np)
bk(4,5)=ak(4,5)*r(kn)/r(np)
bk(4,6)=ak(4,6)*r(kn)/r(np)

bk(5,1)=-ak(5,1)*r(kn)/r(np)
bk(5,2)=-ak(5,2)*r(kn)/r(np)
bk(5,3)=-ak(5,3)*r(kn)/r(np)
bk(5,4)=ak(5,4)*r(kn)/r(np)
bk(5,5)=ak(5,5)*r(kn)/r(np)
bk(5,6)=ak(5,6)*r(kn)/r(np)

bk(6,1)=-ak(6,1)*r(kn)/r(np)
bk(6,2)=-ak(6,2)*r(kn)/r(np)
bk(6,3)=-ak(6,3)*r(kn)/r(np)
bk(6,4)=ak(6,4)*r(kn)/r(np)
bk(6,5)=ak(6,5)*r(kn)/r(np)
bk(6,6)=ak(6,6)*r(kn)/r(np)

a(kkn1,kkn1)=a(kkn1,kkn1)-bk(1,1)
a(kkn1,kkn2)=a(kkn1,kkn2)-bk(1,2)
a(kkn1,kkn3)=a(kkn1,kkn3)-bk(1,3)
a(kkn1,kkn4)=a(kkn1,kkn4)+bk(1,4)*r(kn)
a(kkn1,kkn5)=a(kkn1,kkn5)+bk(1,5)*r(kn)
a(kkn1,kkn6)=a(kkn1,kkn6)+bk(1,6)*r(kn)

a(kkn2,kkn1)=a(kkn2,kkn1)-bk(2,1)
a(kkn2,kkn2)=a(kkn2,kkn2)-bk(2,2)
a(kkn2,kkn3)=a(kkn2,kkn3)-bk(2,3)
a(kkn2,kkn4)=a(kkn2,kkn4)+bk(2,4)*r(kn)
a(kkn2,kkn5)=a(kkn2,kkn5)+bk(2,5)*r(kn)
a(kkn2,kkn6)=a(kkn2,kkn6)+bk(2,6)*r(kn)
```

```
a(kkn3,kkn1)=a(kkn3,kkn1)-bk(3,1)
a(kkn3,kkn2)=a(kkn3,kkn2)-bk(3,2)
a(kkn3,kkn3)=a(kkn3,kkn3)-bk(3,3)
a(kkn3,kkn4)=a(kkn3,kkn4)+bk(3,4)*r(kn)
a(kkn3,kkn5)=a(kkn3,kkn5)+bk(3,5)*r(kn)
a(kkn3,kkn6)=a(kkn3,kkn6)+bk(3,6)*r(kn)

a(kkn4,kkn1)=a(kkn4,kkn1)-bk(4,1)
a(kkn4,kkn2)=a(kkn4,kkn2)-bk(4,2)
a(kkn4,kkn3)=a(kkn4,kkn3)-bk(4,3)
a(kkn4,kkn4)=a(kkn4,kkn4)+bk(4,4)*r(kn)
a(kkn4,kkn5)=a(kkn4,kkn5)+bk(4,5)*r(kn)
a(kkn4,kkn6)=a(kkn4,kkn6)+bk(4,6)*r(kn)

a(kkn5,kkn1)=a(kkn5,kkn1)-bk(5,1)
a(kkn5,kkn2)=a(kkn5,kkn2)-bk(5,2)
a(kkn5,kkn3)=a(kkn5,kkn3)-bk(5,3)
a(kkn5,kkn4)=a(kkn5,kkn4)+bk(5,4)*r(kn)
a(kkn5,kkn5)=a(kkn5,kkn5)+bk(5,5)*r(kn)
a(kkn5,kkn6)=a(kkn5,kkn6)+bk(5,6)*r(kn)

a(kkn6,kkn1)=a(kkn6,kkn1)-bk(6,1)
a(kkn6,kkn2)=a(kkn6,kkn2)-bk(6,2)
a(kkn6,kkn3)=a(kkn6,kkn3)-bk(6,3)
a(kkn6,kkn4)=a(kkn6,kkn4)+bk(6,4)*r(kn)
a(kkn6,kkn5)=a(kkn6,kkn5)+bk(6,5)*r(kn)
a(kkn6,kkn6)=a(kkn6,kkn6)+bk(6,6)*r(kn)

a(kkn1,knp1)=bk(1,1)
a(kkn1,knp2)=bk(1,2)
a(kkn1,knp3)=bk(1,3)
a(kkn1,knp4)=bk(1,4)*r(np)
a(kkn1,knp5)=bk(1,5)*r(np)
a(kkn1,knp6)=bk(1,6)*r(np)

a(kkn2,knp1)=bk(2,1)
a(kkn2,knp2)=bk(2,2)
a(kkn2,knp3)=bk(2,3)
a(kkn2,knp4)=bk(2,4)*r(np)
a(kkn2,knp5)=bk(2,5)*r(np)
a(kkn2,knp6)=bk(2,6)*r(np)

a(kkn3,knp1)=bk(3,1)
a(kkn3,knp2)=bk(3,2)
a(kkn3,knp3)=bk(3,3)
a(kkn3,knp4)=bk(3,4)*r(np)
a(kkn3,knp5)=bk(3,5)*r(np)
a(kkn3,knp6)=bk(3,6)*r(np)

a(kkn4,knp1)=bk(4,1)
a(kkn4,knp2)=bk(4,2)
```

```
                a(kkn4,knp3)=bk(4,3)
                a(kkn4,knp4)=bk(4,4)*r(np)
                a(kkn4,knp5)=bk(4,5)*r(np)
                a(kkn4,knp6)=bk(4,6)*r(np)

                a(kkn5,knp1)=bk(5,1)
                a(kkn5,knp2)=bk(5,2)
                a(kkn5,knp3)=bk(5,3)
                a(kkn5,knp4)=bk(5,4)*r(np)
                a(kkn5,knp5)=bk(5,5)*r(np)
                a(kkn5,knp6)=bk(5,6)*r(np)

                a(kkn6,knp1)=bk(6,1)
                a(kkn6,knp2)=bk(6,2)
                a(kkn6,knp3)=bk(6,3)
                a(kkn6,knp4)=bk(6,4)*r(np)
                a(kkn6,knp5)=bk(6,5)*r(np)
                a(kkn6,knp6)=bk(6,6)*r(np)

                b(knp1)=b(knp1)-fx
                b(knp2)=b(knp2)-fy
                b(knp3)=b(knp3)-fz
                b(knp4)=b(knp4)-mx0
                b(knp5)=b(knp5)-my0
                b(knp6)=b(knp6)-mz0

                b(kkn1)=b(kkn1)+fx
                b(kkn2)=b(kkn2)+fy
                b(kkn3)=b(kkn3)+fz
                b(kkn4)=b(kkn4)-mx0*r(kn)/r(np)
                b(kkn5)=b(kkn5)-my0*r(kn)/r(np)
                b(kkn6)=b(kkn6)-mz0*r(kn)/r(np)

22              continue
21          continue

            sumb=0.0
            do 804 i=1,npt
               iitp=6*(i-1)
               sumb=sumb+abs(b(iitp+1))+abs(b(iitp+2))+abs(b(iitp+3))
804         continue

            if(inner.eq.1)then
               sumb1=sumb
               ratb=1.0
               psumb5=sumb
            end if

            if(mod(inner,5).eq.0)then
               ratb=abs(psumb5-sumb)/sumb1
               psumb5=sumb
            end if
```

```
            ssfx=vec_$asum(sfx,npt)
            ssfy=vec_$asum(sfy,npt)
            ssfz=vec_$asum(sfz,npt)
            ssmx=vec_$asum(smx,npt)
            ssmy=vec_$asum(smy,npt)
            ssmz=vec_$asum(smz,npt)

            rtx=0.0
            rty=0.0
            rtz=0.0
            rtmx=0.0
            rtmy=0.0
            rtmz=0.0
            if(abfx.ne.0.0)rtx=ssfx/(2.0*abfx)
            if(abfy.ne.0.0)rty=ssfy/(2.0*abfy)
            if(abfz.ne.0.0)rtz=ssfz/(2.0*abfz)
            if(abmx.ne.0.0)rtmx=ssmx/(2.0*abmx)
            if(abmy.ne.0.0)rtmy=ssmy/(2.0*abmy)
            if(abmz.ne.0.0)rtmz=ssmz/(2.0*abmz)

            volume=ncell*delx*dely*delz
            if(mod(inner,5).eq.0)then
               call stress(dm,npt,monop,epsa,nx,ny,nz,fn1,ft1,r
     +           ,volume,s,sigma,0,isigm)
            end if

c            write(6,48)inner,rtx,rty,rtz,sumb,ratb,nbon,kslp,sigma
c            if(ist.eq.idist1*(ist/idist1))then
c               write(49,48)inner,rtx,rty,rtz,sumb,ratb,nbon,kslp,sigma
c 48            format(5x,i5,3f8.3,2f10.5,2i5,f12.8)
c            end if

         lfl=0
         if(iprepk.le.1)then
            if(inner.ge.10.and.sigma.lt.sigma0)lfl=1
            if(inner.ge.10.and.(rtx+rty+rtz).lt.0.03)lfl=1
         else
            if(cs.ne.0.0)then
               if(sumb.lt.epsi*0.5.or.ratb.lt.epsi*0.5.or.(rtx+rty+rtz+
     +         rtmx+rtmy+rtmz).lt.0.42)lfl=1
            else
               if(sumb.lt.epsi.or.ratb.lt.epsi.or.(rtx+rty+rtz).lt.
     +         0.21)lfl=1
            end if
            if(sigma.lt.0.5*sigma0.and.inner.ge.10)lfl=1
         end if
         if(inner.gt.inmaxo)lfl=1

            if(lfl.eq.0)then
               go to 666
            else
```

```
            if(iway.eq.2.or.abs(amda).lt.0.00001)then
          ckn=1.0
          ckt=0.8

          fnav_cur=fnav_cur/real(nbon)

c         write(*,*)'fnaverage',fnaverage
c         write(*,*)'fnav_cur',fnav_cur
                go to 888
            else
          iway=2
          fnaverage=fnaverage/real(nbon)
          fnav_cur=fnav_cur/real(nbon)

          inner=1
          go to 333
            end if
        end if

 666    if(iprepk.gt.1.and.sigma.lt.0.5*sigma0.and.mod(inner,5).eq.0)then
              go to 814
       end if

            if(iprepk.le.1.and.density.lt.0.5)then
               epsdo=epsi*10.0
               irelo=irel*0.5
            else
               epsdo=epsi
               irelo=irel
            end if

c---solving the quasi-linear system of equations

            call relax(idim,npt,irelo,epsdo,orf,a,b,xn,numr,res,
     +              dxn,ist,iterout,inner)

c---updating the fluctuation displacement

            do 382 itm=1,npt
               ktm=3*(dm-1)*(itm-1)
               ux(itm)=ux(itm)+xn(ktm+1)
               uy(itm)=uy(itm)+xn(ktm+2)
               uz(itm)=uz(itm)+xn(ktm+3)
               wx(itm)=wx(itm)+xn(ktm+4)
               wy(itm)=wy(itm)+xn(ktm+5)
               wz(itm)=wz(itm)+xn(ktm+6)
 382        continue

            go to 333

c---once force balance is achieved, computing the stress
```

```
 888        call stress(dm,npt,monop,epsa,nx,ny,nz,fn1,ft1,r
     +        ,volume,s,sigma,1,isigm)

c          if(ist.eq.idist1*(ist/idist1))then
c          write(49,47)inner,rtx,rty,rtz,rtmx,rtmy,rtmz,nbon,kslp,sigma
c 47            format(1x,i4,6f8.2,2i5,f10.7)
c             end if

c         write(98,649)ist,iterout,inner,numr,rtx,rty,rtz,rtmx,rtmy,rtmz
c 649            format(3x,3i5,i7,6f8.3)

 814          test=sigma-sigma0
              if(test.lt.0.0)then
                 stest=-1
              else
                 stest=1
              end if
              error=test/sigma0

              if(iter2.eq.2.and.abs(error).lt.errormin)then
                 delevmin=delev
                 errormin=abs(error)
              end if

c       write(50,601)ir,ist,iterout,stest,strn(3,3),
c      +        error,delev,ev,density
        if(iprepk.le.1)then
           write(*,*)'ir=',ir,',   trial=',itrial,',   ist=',ist,
     +        ',   density=',density
        else if(iprepk.eq.2)then
c          write(*,*)'I am relaxing the stress, please wait.'
        end if
       write(*,601)ir,ist,iterout,stest,test,error,delev,ev,density,isigm
 601          format(i3,i5,2i3,f12.9,f10.6,f13.9,2f10.6,i5)

c---check if packing criteria are met or not

          if(iprepk.le.1.and.density.gt.den0.and.error.le.epso.
     +    and.abs(sstrn(3,1)+sstrn(1,2)).lt.1.0e-8)kflag=1

          return
          end
c-------------------------------------------------------------------------
c---mapping particles into microcell coordinate system

       subroutine mapping(npt,nm,mxnc,sstrn,delx,dely,delz,x,y,z,map,
      +  mappart,iouts)
       implicit real (a-h,o-z)
       implicit integer (i-n)

       real sstrn(3,3),x(npt),y(npt),z(npt),delx,dely,delz,dx,dy,dz
       integer map(npt),mappart(mxnc),nm
```

```
dx=real(nm)*delx
dy=real(nm)*dely
dz=real(nm)*delz
iouts=0

call vec_$iinit(mappart,mxnc,0)

do 1 i=1,npt

   xmin=tan(sstrn(2,1))*y(i)+tan(sstrn(3,1))*z(i)
   if(x(i).ge.xmin)then
      mx=int((x(i)-xmin)/dx)
   else
      mx=int((x(i)-xmin)/dx)-1
   end if

   ymin=tan(sstrn(1,2))*x(i)+tan(sstrn(3,2))*z(i)
   if(y(i).ge.ymin)then
      my=int((y(i)-ymin)/dy)
   else
      my=int((y(i)-ymin)/dy)-1
   end if

   zmin=tan(sstrn(1,3))*x(i)+tan(sstrn(2,3))*y(i)
   if(z(i).ge.zmin)then
      mz=int((z(i)-zmin)/dz)
   else
      mz=int((z(i)-zmin)/dz)-1
   end if

   x(i)=x(i)-mx*dx-my*tan(sstrn(2,1))*dy-mz*tan(sstrn(3,1))*dz
   y(i)=y(i)-my*dy-mx*tan(sstrn(1,2))*dx-mz*tan(sstrn(3,2))*dz
   z(i)=z(i)-mz*dz-mx*tan(sstrn(1,3))*dx-my*tan(sstrn(2,3))*dy

   xmin=tan(sstrn(2,1))*y(i)+tan(sstrn(3,1))*z(i)
   ymin=tan(sstrn(1,2))*x(i)+tan(sstrn(3,2))*z(i)
   zmin=tan(sstrn(1,3))*x(i)+tan(sstrn(2,3))*y(i)

   kb=int((z(i)-zmin)/delz)+1
   if(kb.eq.nm+1)kb=nm
   jb=int((y(i)-ymin)/dely)+1
   if(jb.eq.nm+1)jb=nm
   ib=int((x(i)-xmin)/delx)+1
   if(ib.eq.nm+1)ib=nm
   ibar=ib+(jb-1)*nm+(kb-1)*nm*nm

   if(abs(mx).ge.2.and.abs(my).ge.2.and.abs(mz).ge.2)then
      write(*,*)'particle  ',i,'  is located outside'
      write(*,*)'kb,jb,ib,nm,ibar=  ',kb,jb,ib,nm,ibar
      iouts=iouts+1
   end if
```

```
         map(i)=ibar
         mappart(ibar)=i
 1    continue

      return
      end
c-------------------------------------------------------
c---computing microcell adjacency matrix and "kxx,kyy,kzz"

      subroutine adj3(mxnc,ijs,nm,nnb,adjc,kxx,kyy,kzz)
      implicit real (a-h,o-z)
      implicit integer (i-n)

      integer adjc(mxnc,nnb)
      integer kxx(mxnc,nnb),kyy(mxnc,nnb),kzz(mxnc,nnb)

      do 60 k=1,nm
       do 50 j=1,nm
         do 40 i=1,nm
            k1=i+(j-1)*nm+(k-1)*nm*nm
            index=0
            do 31 kk=k-ijs,k+ijs
               if(kk.lt.1)then
                  kz=1
               else if(kk.gt.nm)then
                  kz=-1
               else
                  kz=0
               end if
               kkbar=kk+kz*nm

            do 35 kj=j-ijs,j+ijs
               if(kj.lt.1)then
                  ky=1
               else if(kj.gt.nm)then
                  ky=-1
               else
                  ky=0
               end if
               kjbar=kj+ky*nm

               do 30 ki=i-ijs,i+ijs
                  if(ki.lt.1)then
                     kx=1
                  else if(ki.gt.nm)then
                     kx=-1
                  else
                     kx=0
                  end if
                  kibar=ki+kx*nm
```

```
          k2=kibar+(kjbar-1)*nm+(kkbar-1)*nm*nm
          if(k1.eq.k2)go to 30
          index=index+1
                  adjc(k1,index)=k2
                  kxx(k1,index)=kx
                  kyy(k1,index)=ky
                  kzz(k1,index)=kz
 30            continue
 35          continue
 31       continue
 40       continue
 50       continue
 60     continue

        return
        end

c-------------------------------------------------------
c---linear equation solver

        subroutine relax(idim,npr,irel,epsd,orf,a,b,xn,nr,res,dxn,
       +       ist,iout,inn)
        implicit real (a-h,o-z)
        implicit integer (i-n)

        real a(idim,idim),b(idim),res(idim),dxn(idim),xn(idim)

        zero=0.0
        call vec_$init(xn,idim,zero)
        call vec_$init(dxn,idim,zero)
        call vec_$init(res,idim,zero)

        inum=0
 100    inum=inum+1

        rmax=0.0
        imax=0
        resum=0.0
        if(inum.gt.1)then
           do 22 i=1,idim
              if(abs(a(i,jmax)).lt.1.0e-10)go to 21
              res(i)=res(i)+a(i,jmax)*dxn(jmax)
 21           resum=resum+abs(res(i))
              if(abs(res(i)).gt.rmax)then
                 rmax=abs(res(i))
                 imax=i
              end if
 22        continue

        else
           do 20 i=1,idim
              if(abs(b(i)).lt.1.0e-12)go to 20
```

```
            res(i)=-b(i)
            resum=resum+abs(res(i))
            if(abs(res(i)).gt.rmax)then
               rmax=abs(res(i))
               imax=i
            end if
 20      continue
         resum1=resum
         presum50=resum

      end if

      jmax=imax
      dxn(jmax)=-res(imax)/a(imax,jmax)
      xn(jmax)=xn(jmax)+dxn(jmax)

      if(mod(inum,50).eq.0)then
         rati50=abs(presum50-resum)/resum1
         presum50=resum
         if(rati50.gt.epsd.and.inum.lt.irel)go to 100
         nr=inum
         return
      end if
      go to 100

      end

c-------------------------------------------------
c---shuffling algorithm

      subroutine shuffle(ncell,nshuf,lflip,nrif,iseed,m,m1,k1,k2)

      implicit real (a-h,o-z)
      implicit integer (i-n)

      integer m(ncell),m1(ncell),k1(ncell),k2(ncell)

         do 65 i=1,ncell
            m(i)=i
            m1(i)=0
            k1(i)=0
            k2(i)=0
 65      continue

      nce1=ncell/2
      nce2=ncell-nce1

      do 100 jim=1,nshuf

         do 105 lf=1,lflip
            mid=int(ran(iseed)*ncell)
            do 102 i=1,ncell
```

```
               if(i.le.ncell-mid)then
                  m1(i)=m(mid+i)
               else
                  m1(i)=m(i-ncell+mid)
               end if
 102        continue
            call vec_$icopy(m1,m,ncell)
 105     continue

         do 500 ir=1,nrif
            do 110 i=1,nce1
               k1(i)=m(i)
 110        continue
            do 120 i=1,nce2
               k2(i)=m(nce1+i)
 120        continue

            do 130 i=nce2,1,-1
               m(2*i-1)=k2(i)
 130        continue
            do 140 i=nce1,1,-1
               m(2*i)=k1(i)
 140        continue
 500     continue

 100  continue
      return
      end

c-------------------------------------------------------------
c---initialization of a real 2D array

      subroutine rinit2(irow,icol,a,value)

      real a(irow,icol),value

      do 5 j=1,icol
         do 10 i=1,irow
            a(i,j)=value
 10      continue
 5    continue
      return
      end
c-------------------------------------------------------------
c---initialization of a real 3D array

      subroutine rinit3(irow,icol,lay,a,value)

      real a(irow,icol,lay),value

      do 15 l=1,lay
         do 5 j=1,icol
```

```
            do 10 i=1,irow
               a(i,j,1)=value
 10         continue
 5       continue
 15    continue
       return
       end
c-------------------------------------------------
c---initialization of an integer 2D array

      subroutine init2(irow,icol,ia,value)

      integer ia(irow,icol),value

      do 5 j=1,icol
         do 10 i=1,irow
            ia(i,j)=value
 10      continue
 5     continue
       return
       end
c-------------------------------------------------
c---copy from one integer 2D array to another

      subroutine matx_copy(irow,icol,iu,iv)

      integer iu(irow,icol),iv(irow,icol)

      do 5 j=1,icol
         do 10 i=1,irow
            iv(i,j)=iu(i,j)
 10      continue
 5     continue
       return
       end
c-------------------------------------------------
c---copy from one real 2D array to another

      subroutine rmatx_copy(irow,icol,u,v)

      real u(irow,icol),v(irow,icol)

      do 5 j=1,icol
         do 10 i=1,irow
            v(i,j)=u(i,j)
 10      continue
 5     continue
       return
       end
c-------------------------------------------------
c---copy from one real 3D array to another
```

```
      subroutine rmatx3_copy(irow,icol,lay,u,v)

      real u(irow,icol,lay),v(irow,icol,lay)

      do 15 l=1,lay
         do 5 j=1,icol
            do 10 i=1,irow
               v(i,j,l)=u(i,j,l)
 10         continue
 5       continue
 15   continue
      return
      end
c-------------------------------------
c---subroutine to choose radii randomly for poly-disperse system

      subroutine r_assign(n1,n2,n3,n10,n20,n30,r1,r2,r3,p1,p2,iseed,r)

      implicit real (a-h,o-z)
      implicit integer (i-n)

      if((n1.lt.n10).and.(n2.lt.n20).and.(n3.lt.n30))then
         ra=ran(iseed)
         if(ra.lt.p1)then
            r=r1
            n1=n1+1
         else if((ra.gt.p1).and.(ra.lt.p2))then
            r=r2
            n2=n2+1
         else
            r=r3
            n3=n3+1
         end if
      else if((n2.lt.n20).and.(n3.lt.n30))then
         ra=ran(iseed)
         if(ra.lt.(p2-p1)/(1.0-p1))then
            r=r2
            n2=n2+1
         else
            r=r3
            n3=n3+1
         end if
      else if((n1.lt.n10).and.(n2.lt.n20))then
         ra=ran(iseed)
         if(ra.lt.p1/p2)then
            r=r1
            n1=n1+1
         else
            r=r2
            n2=n2+1
         end if
      else if((n1.lt.n10).and.(n3.lt.n30))then
```

```
            ra=ran(iseed)
            if(ra.lt.p1/(1.0-p2+p1))then
               r=r1
               n1=n1+1
            else
               r=r3
               n3=n3+1
            end if
         else
            if(n3.lt.n30)then
               r=r3
               n3=n3+1
            else if(n2.lt.n20)then
               r=r2
               n2=n2+1
            else
               r=r1
               n1=n1+1
            end if
         end if

         return
         end

c-----------------------------------------------
c---calculating stress tensor

         subroutine stress(dm,npt,monop,epsa,nx,ny,nz,fn1,ft1,r,
     +   volume,s,sigma,i0or1,isigm)

         implicit real (a-h,o-z)
         implicit integer (i-n)

         integer dm
         real nx(npt,npt),ny(npt,npt),nz(npt,npt),fn1(npt,npt)
         real nxx,nyy,nzz,r(npt),ft1(npt,npt,dm),s(dm,dm)

            do 877 i=1,npt-1
               do 875 j=i+1,npt
                  nx(j,i)=-nx(i,j)
                  ny(j,i)=-ny(i,j)
                  nz(j,i)=-nz(i,j)
                  fn1(j,i)=fn1(i,j)
           do 874 k=1,dm
                     ft1(j,i,k)=-ft1(i,j,k)
874        continue
875             continue
877        continue

           do 10 i=1,dm
              do 15 j=1,dm
           s(j,i)=0.0
```

```
15          continue
10        continue

          if(monop.eq.1)then
             do 121 np=1,npt-1
                do 122 kn=np+1,npt
                   if(fn1(np,kn).ge.-epsa)go to 122
                   nxx=nx(np,kn)
                   nyy=ny(np,kn)
                   nzz=nz(np,kn)
           fn1x=fn1(np,kn)*nxx
           fn1y=fn1(np,kn)*nyy
           fn1z=fn1(np,kn)*nzz
                   s(1,1)=s(1,1)+r(np)*nxx*(fn1x+ft1(np,kn,1))
                   s(2,2)=s(2,2)+r(np)*nyy*(fn1y+ft1(np,kn,2))
                   s(3,3)=s(3,3)+r(np)*nzz*(fn1z+ft1(np,kn,3))

           if(i0or1.eq.1)then
                     s(1,2)=s(1,2)+r(np)*nxx*(fn1y+ft1(np,kn,2))
                     s(1,3)=s(1,3)+r(np)*nxx*(fn1z+ft1(np,kn,3))
                     s(2,1)=s(2,1)+r(np)*nyy*(fn1x+ft1(np,kn,1))
                     s(2,3)=s(2,3)+r(np)*nyy*(fn1z+ft1(np,kn,3))
                     s(3,1)=s(3,1)+r(np)*nzz*(fn1x+ft1(np,kn,1))
                     s(3,2)=s(3,2)+r(np)*nzz*(fn1y+ft1(np,kn,2))

           end if

122             continue
121          continue

          else
             do 721 np=1,npt
                do 722 kn=1,npt
                   if(fn1(np,kn).ge.-epsa.or.np.eq.kn)go to 722
                   nxx=nx(np,kn)
                   nyy=ny(np,kn)
                   nzz=nz(np,kn)
           fn1x=fn1(np,kn)*nxx
           fn1y=fn1(np,kn)*nyy
           fn1z=fn1(np,kn)*nzz
                   s(1,1)=s(1,1)+r(np)*nxx*(fn1x+ft1(np,kn,1))
                   s(2,2)=s(2,2)+r(np)*nyy*(fn1y+ft1(np,kn,2))
                   s(3,3)=s(3,3)+r(np)*nzz*(fn1z+ft1(np,kn,3))
           if(i0or1.eq.1)then
                     s(1,2)=s(1,2)+r(np)*nxx*(fn1y+ft1(np,kn,2))
                     s(1,3)=s(1,3)+r(np)*nxx*(fn1z+ft1(np,kn,3))
                     s(2,1)=s(2,1)+r(np)*nyy*(fn1x+ft1(np,kn,1))
                     s(2,3)=s(2,3)+r(np)*nyy*(fn1z+ft1(np,kn,3))
                     s(3,1)=s(3,1)+r(np)*nzz*(fn1x+ft1(np,kn,1))
                     s(3,2)=s(3,2)+r(np)*nzz*(fn1y+ft1(np,kn,2))

           end if
```

```
722              continue
721            continue

               end if

               do 20 i=1,dm
                  do 25 j=1,dm
             if(monop.eq.1)s(j,i)=2.0*s(j,i)/volume
             if(monop.eq.2)s(j,i)=s(j,i)/volume
25               continue
20             continue

               if(isigm.eq.2)then
                  sigma=-(s(1,1)+s(2,2))/2.0
               else
                  sigma=-(s(1,1)+s(2,2)+s(3,3))*0.333333
               end if

               return
               end
c--------------------------------------

        subroutine conduct(npt,mxnc,nnb,dx,dy,dz,sstrn,x,y,z,r,
     +     nx,ny,nz,cond,brx,bry,brz,cur,voltf,c_eff,
     +     cxn,cres,cdxn,map,mappart,adjc,kxx,kyy,kzz,grad,barR)

        real sstrn(3,3),x(npt),y(npt),z(npt),r(npt)
        real nx(npt,npt),ny(npt,npt),nz(npt,npt)
        real cond(npt,npt),brx(npt),bry(npt),brz(npt)
        real cur(npt,npt),voltf(npt),c_eff(3,3)
        real cxn(npt),cres(npt),cdxn(npt)
        integer map(npt),kxx(mxnc,nnb),kyy(mxnc,nnb),kzz(mxnc,nnb)
        integer adjc(mxnc,nnb),mappart(mxnc)

            call rinit2(3,3,c_eff,0.0)

            call relax_v(npt,cond,brx,cxn,cres,cdxn)
            call vec_$copy(cxn,voltf,npt)
c           write(60,*)ir,ist,' x'

            fx=0.0
            fy=0.0
            fz=0.0
            do 41 np=1,npt
c              sum=0.0
               do 42 j_kn=1,nnb

          kn=mappart(adjc(map(np),j_kn))
          if(kn.eq.0)go to 42

          cur(np,kn)=0.0
```

```
          if(abs(cond(np,kn)).lt.1.0e-15.or.np.eq.kn)go to 42

          xl=kxx(map(np),j_kn)*dx+kyy(map(np),j_kn)*tan(sstrn(2,1))*dy
     +      +kzz(map(np),j_kn)*tan(sstrn(3,1))*dz

                xkn=x(kn)-xl
           cur(np,kn)=cond(np,kn)*((x(np)-xkn)*barR*grad
     +                +voltf(np)-voltf(kn))
c          write(58,*)cur(np,kn)
c          sum=sum+cur(np,kn)
           fx=fx+r(np)*barR*nx(np,kn)*cur(np,kn)
           fy=fy+r(np)*barR*ny(np,kn)*cur(np,kn)
           fz=fz+r(np)*barR*nz(np,kn)*cur(np,kn)
 42            continue
c              write(60,*)np,sum
 41        continue

           fx=fx/(dx*dy*dz*barR*barR*barR)
           fy=fy/(dx*dy*dz*barR*barR*barR)
           fz=fz/(dx*dy*dz*barR*barR*barR)
           c_eff(1,1)=-fx
           c_eff(1,2)=-fy
           c_eff(1,3)=-fz

           call relax_v(npt,cond,bry,cxn,cres,cdxn)
           call vec_$copy(cxn,voltf,npt)
c          write(60,*)ir,ist,' y'

           fx=0.0
           fy=0.0
           fz=0.0
           do 51 np=1,npt
c             sum=0.0
              do 52 j_kn=1,nnb

           kn=mappart(adjc(map(np),j_kn))
           if(kn.eq.0)go to 52

           cur(np,kn)=0.0
           if(abs(cond(np,kn)).lt.1.0e-15.or.np.eq.kn)go to 52

           yl=kyy(map(np),j_kn)*dy+kxx(map(np),j_kn)*tan(sstrn(1,2))*dx
     +       +kzz(map(np),j_kn)*tan(sstrn(3,2))*dz

                ykn=y(kn)-yl
           cur(np,kn)=cond(np,kn)*((y(np)-ykn)*barR*grad
     +                +voltf(np)-voltf(kn))
c          sum=sum+cur(np,kn)
           fx=fx+r(np)*barR*nx(np,kn)*cur(np,kn)
           fy=fy+r(np)*barR*ny(np,kn)*cur(np,kn)
           fz=fz+r(np)*barR*nz(np,kn)*cur(np,kn)
 52            continue
```

```
c              write(60,*)np,sum
 51          continue

             fx=fx/(dx*dy*dz*barR*barR*barR)
             fy=fy/(dx*dy*dz*barR*barR*barR)
             fz=fz/(dx*dy*dz*barR*barR*barR)
             c_eff(2,1)=-fx
             c_eff(2,2)=-fy
             c_eff(2,3)=-fz

             call relax_v(npt,cond,brz,cxn,cres,cdxn)
             call vec_$copy(cxn,voltf,npt)
c            write(60,*)ir,ist,' z'

             fx=0.0
             fy=0.0
             fz=0.0

c            c_top=0.0
c            c_bot=0.0

             do 61 np=1,npt
c               sum=0.0
                do 62 j_kn=1,nnb

           kn=mappart(adjc(map(np),j_kn))
           if(kn.eq.0)go to 62

           cur(np,kn)=0.0
           if(abs(cond(np,kn)).lt.1.0e-15.or.np.eq.kn)go to 62

            zl=kzz(map(np),j_kn)*dz+kxx(map(np),j_kn)*tan(sstrn(1,3))*dx
     +        +kyy(map(np),j_kn)*tan(sstrn(2,3))*dy

                 zkn=z(kn)-zl
            cur(np,kn)=cond(np,kn)*((z(np)-zkn)*barR*grad
     +                +voltf(np)-voltf(kn))
c-------------
c          if(kzz(map(np),map(kn)).eq.1)c_top=c_top+cur(np,kn)
c          if(kzz(map(np),map(kn)).eq.-1)c_bot=c_bot+cur(np,kn)
c-------------
c          if(np.le.10)write(60,*)np,kn,cur(np,kn)
c          sum=sum+cur(np,kn)
           fx=fx+r(np)*barR*nx(np,kn)*cur(np,kn)
           fy=fy+r(np)*barR*ny(np,kn)*cur(np,kn)
           fz=fz+r(np)*barR*nz(np,kn)*cur(np,kn)
 62             continue
c               write(60,*)np,sum

c               write(59,*)z(np),z(np)*gradp+voltf(np)

 61          continue
```

```
            fx=fx/(dx*dy*dz*barR*barR*barR)
            fy=fy/(dx*dy*dz*barR*barR*barR)
            fz=fz/(dx*dy*dz*barR*barR*barR)
            c_eff(3,1)=-fx
            c_eff(3,2)=-fy
            c_eff(3,3)=-fz

c---------------
c           write(95,*)c_top,c_bot
c           write(95,*)dz*barR*c_top/(dx*barR*dy*barR*grad*dz*barR)
c           write(95,*)c_eff(3,3)

c---------------

        return
        end


c-------------------------------------------------------
c---linear equation solver

      subroutine relax_v(npt,cond,br,cxn,cres,cdxn)
      implicit real (a-h,o-z)
      implicit integer  (i-n)

      real cond(npt,npt),br(npt),cxn(npt),cres(npt),cdxn(npt)

      zero=0.0
      call vec_$init(cxn,npt,zero)
      call vec_$init(cdxn,npt,zero)
      call vec_$init(cres,npt,zero)

      inum=0
 100  inum=inum+1

      rmax=0.0
      imax=0
      cresum=0.0
      if(inum.gt.1)then
         do 22 i=1,npt
            if(abs(cond(i,jmax)).lt.1.0e-15)go to 21
            cres(i)=cres(i)+cond(i,jmax)*cdxn(jmax)
 21         cresum=cresum+abs(cres(i))
            if(abs(cres(i)).gt.rmax)then
               rmax=abs(cres(i))
               imax=i
            end if
 22      continue
      else
         do 20 i=1,npt
            if(br(i).eq.0.0)go to 20
```

```
            cres(i)=-br(i)
            cresum=cresum+abs(cres(i))
            if(abs(cres(i)).gt.rmax)then
               rmax=abs(cres(i))
               imax=i
            end if
 20       continue
          cresum1=cresum
       end if

       jmax=imax

       cdxn(jmax)=-cres(imax)/cond(imax,jmax)
       cxn(jmax)=cxn(jmax)+cdxn(jmax)

       if(cresum.gt.cresum1*0.00001.and.inum.lt.10000)go to 100
c       write(60,*)'cresum1,cresum,inum',cresum1,cresum,inum

       return
       end

c-------------------------------------------------
c---copy one half of a 2D array(above diagnol) to another half

       subroutine copyhalf(irow,u)

       real u(irow,irow)

       do 5 i=1,irow-1
          do 10 j=i+1,irow
             u(j,i)=u(i,j)
 10       continue
 5     continue
       return
       end

c-------------------------------------------------------

       subroutine degree_phi(nz,kphi)
       implicit real (a-h,o-z)
       implicit integer  (i-n)

       real nz
       integer kphi(12)

       dnz=0.1666667
       if(nz.gt.0.0)then
          k=6-int(nz/dnz)
       else
          k=-int(nz/dnz)+7
       end if
       kphi(k)=kphi(k)+1
```

```
      return
      end

c-----------------------------------------------------
      subroutine degree_theta(nnx,nny,ktheta)
      implicit real (a-h,o-z)
      implicit integer  (i-n)

      real nx,ny,nnx,nny
      integer  ktheta(24)

      pi=3.141593

      nx=nnx/sqrt(nnx*nnx+nny*nny)
      ny=nny/sqrt(nnx*nnx+nny*nny)

      if(ny.ge.0.0)then
         if(nx.ge.0.0)then
            theta=asin(abs(ny))
         else
            theta=pi-asin(abs(ny))
         end if
      else
         if(nx.ge.0.0)then
            theta=2.0*pi-asin(abs(ny))
         else
            theta=pi+asin(abs(ny))
         end if
      end if

      theta=(theta/pi)*180.0
      k=int(theta/15.0)+1
      ktheta(k)=ktheta(k)+1

      return
      end
```

Bibliography

[1] Agarwal, T.K., Micromechanics of Granular Materials and Its Relation to Wave Velocity, Ph.D. dissertation, Old Dominion University, Norfolk (1991).

[2] Agarwal, T.K. and Ishibashi, I., Anisotropic Elastic Constants of Granular Assembly From Wave Velocity measurements, *Advances in Micromechanics of Granular Materials*, eds. Shen, H.H. et al., pp. 51-60, Elsevier, Amsterdam (1992).

[3] Allen, M.P. and Tildesley, D.J., *Computer Simulation of Liquids*, Clarendon, Oxford, 1987.

[4] Anandarajah, A., In Situ prediction of Stress-Strain Relationships of Clays Using a Bounding Surface Plasticity Model and an Electrical Method, Ph.D. dissertation, University of California, Davis, (1982).

[5] Anandarajah, A., Sobhan, K. and Kuganenthira, N., Fabric Anisotropy and Incremental Stress-Strain Behavior of Soils, *Proc. of the 2nd International Conference on Discrete Element Methods*, eds. J.R. Williams and Graham G.W. Mustoe, pp. 547-556 (1993).

[6] Antler, M., Effect of Surface Contamination on Electric Contact Performance, *IEEE Circuits and Devices Magazine*, **3**, pp.8-20, (1987).

[7] Archard, J.F., Elastic Deformation and the Laws of Friction, *Proc. Roy. Soc. London*, **A243**, pp.190-205, (1957).

[8] Arulanandan, K. and Kutter, B., A Directional Structure Index Related to Sand Liquefaction, *Proc. Spec. Conf. Earthquake Eng. Soil Dynamics, ASCE*, Pasadena, California, June 19-21, 1978, pp.213-230.

[9] Bala, K., Pradhan, P.R., Saxena, N.S. and Saksena, M.P., Effective Thermal Conductivity of Copper Powders, *J. Physics D, Appl. Physics*, **22**, pp.1068-1072, (1989).

[10] Bardet, J.P. and Proubet, J., An Adaptive Relaxation Technique for the Statics of Granular materials, *Computers and Structures*, **39**, No.3/4, pp.221-229 (1991).

[11] Bardet, J.P. and Proubet, J., A Numerical Investigation of the Structure of Persistent Shear Bands in Granular Media, *Geotechnique*, **41**, No.4, pp.599-613 (1992).

[12] Bardet, J.P. and Proubet, J., A Shear Band Analysis in Idealized Granular Material, *J. of Engineering Mechanics, ASCE*, **118**, No.2, pp.397-415 (1992).

[13] Bashir, Y.M. and Goddard, J.D., Experiments on the Conductivity of Suspensions of Ionically-Conductive Spheres, *AICHE Journal*, **36**, No.3, pp.387-396, (1990).

[14] Bashir, Y.M. and Goddard, J.D., A Novel Simulation Method for the Quasi-static Mechanics of Granular Assemblages , *J. Rheology*, **35**, pp. 849-885 (1991).

[15] Bashir, Y.M., Mechanics and Transport in Granular Media, Ph.D. dissertation, University of Southern California, Los Angeles (1990).

[16] Batchelor, G.K. and O'Brien, R.W., Thermal or Electrical Conduction through a Granular material, *Proc. R. Soc. Lond. A.*, **355**, pp. 313-333 (1977).

[17] Bathurst, R.J. and Rothenburg, L., Micromechanical Aspects of Isotropic Granular Assemblies with Linear Contact Interactions, *J. Appl. Mech.*, **55**, pp.17-23, (1988).

[18] Bathurst, R.J. and Rothenburg, L., Observations of Stress-Force-Fabric Relationships in Idealized granular materials, *Mech. Mater.*, **9**, pp. 65-80 (1990).

[19] Berryman, J.G., Random Close Packing of Hard Spheres and Disks, *Phys. Rev. A*, **27**, pp.1053-1061 (1983).

[20] Boerner, S.T. and Sclater, J.G., Deformation under Extension of Assemblies of Steel Balls in Contact: Application to Sand Box Models, *J. of Geophysical Research*, **97**, No.B4, pp.4969-4990 (1992).

[21] Boltz, R.E. and Tuve, G.L., CRC Handbook of the tables for Applied Engineering Science, CRC Press Inc., (1985).

[22] Bonnecaze, R.T. and Brady, J.F., A Method for Determining the Effective Conductivity of Dispersions of Particles, *Proc. Roy. Soc. London*, **A430**, pp. 285-313, (1990).

[23] Brace, W.F., Electrical Resistivity Changes in Saturated Rocks during Fracture and Frictional Sliding, *J. Geophysical Research*, **73**, No.4, pp.1433-1445, (1968).

[24] Bryant, M.D. and Jin, M, Timewise Increases in Contact Resistance due to Surface Roughness and Corrosion, *Proc. of the Thirty-sixth IEEE Holm Conference on Electric Contacts and the Fifteenth International Conference on Electric Contacts*, Aug. 20-24, 1990, Montreal, Quebec, pp.635-645.

[25] Caven, R.W. and Jalali, J., Predicting the Contact Resistance Distribution of Electrical Contacts by Modeling the Contact Interface, *Proc. of the Thirty-seventh IEEE Holm Conference on Electric Contacts*, Oct. 6-9, 1991, Chicago, pp.83-89.

[26] Chang, C.S. and Misra, A., Computer Simulation and Modeling of Mechanical Properties of Particulates, *Computer and Geotechnics*, **7**, pp. 269-287 (1989).

[27] Chang, C.S., Misra, A. and Xue, J.H., Incremental Stress-Strain Relationships for Regular Packings Made of Multi-sized Particles, *Int. J. Solids Structures*, **25**, pp.665-681, (1989).

[28] Chang, C.S., Sundaram, S.S. and Misra, A., Initial Moduli of Particulated Mass with Frictional Contacts, *Intl. J. for Numerical and Analytical Methods in Geomechanics*, **13**, pp.629-644 (1989).

[29] Chang, C.S., Micromechanics Modeling for Deformation and Failure of Granular Material, *Advances in Micromechanics of Granular Materials*, eds. Shen, H.H. et al., pp. 251-260, Elsevier, Amsterdam (1992).

[30] Chang, C.S., Accuracy and Stability for Static Analysis Using Dynamic Formulation in Discrete Element Methods, *Proc. of the 2nd International Conference on Discrete Element Methods*, eds. J.R. Williams and Graham G.W. Mustoe, pp. 379-389 (1993).

[31] Chen, Y-C., Experimental Determination of Fabric for Granular Materials, Ph.D. dissertation, Cornell University, Ithaca (1986).

[32] Chen, Y-C., Ishibashi, I. and Jenkins, J.T., Dynamic Shear Modulus and Fabric: Part I, Depositional and Induced Anisotropy, *Geotechnique*, **38**, pp.25-32, (1988).

[33] Chen, Y-C., Effect of Inter-particle Friction and Initial Fabric on Fabric Evolution, *J. of the Chinese Institute of Engineers*, **13**, No.2, pp. 147-156 (1990).

[34] Christoffersen, J., Mehrabadi, M.M. and Nemat-Nasser, S., A Micromechanical Description of Granular Material Behavior, *J. Appl. Mech.*, **48**, No.2, pp.339-344, (1981).

[35] Chung, Y.C. and Leal, L.G., An Experimental Study of the Effective Thermal Conductivity of a Sheared Suspension of Rigid Spheres, *Intl. J. Multiphase Flow*, **8**, pp.605-625, (1982).

[36] Cowin, S.C., The Relationship between the Elasticity Tensor and the Fabric Tensor, *Mechanics of Materials*, **4**, pp. 137-147 (1985).

[37] Cowin, S.C., Fabric Dependence of an Anisotropic Strength Criterion, *Mechanics of Materials*, **5**, pp. 251-260 (1986).

[38] Cumberland, D.J. and Crawford, R.J., The Packing of Particles, Handbook of Powder Technology, **6**, Elsevier, (1987).

[39] Cundall, P.A. and Strack, O.D.L., The Distinct Element Method as a Tool for Research in Granular Media, Part I(1978) & II(1979), *NSF Report*, Dept. of Civil and Mineral Eng., U. of Minnesota.

[40] Cundall, P.A. and Strack, O.D.L., Modeling of Microscopic Mechanisms in Granular material, *Mechanics of Granular Materials: New Models and Constitutive Relations*, eds. Jenkins, J.T. and Satake, M., pp. 137-149, Elsevier, Amsterdam (1983).

[41] Cundall, P.A., Computer Simulations of Dense Sphere Assemblies, *Micromechanics of Granular Materials*, eds. Satake, M. and Jenkins, J.T., pp. 113-123, Elsevier, Amsterdam (1987).

[42] Cundall, P.A., Numerical Experiments on Localization in Frictional Materials, *Ingenieur-Archiv*, **59**, pp. 148-159 (1989).

[43] Diaconis, P., *Group Representations in Probability and Statistics*, Institute of Mathematical Statistics, Hayward, California, 1988.

[44] Digby, P.J., The effective Elastic Moduli of Porous Granular Rock, *J. of Appl. Mech.*, ASME, **48**, No.4, pp. 803-808 (1981).

[45] Drescher, A., Photoelastic Verification of a Mechanical Model for the Flow of a Granular Material, *J. Mech. Phys. Solids*, **20**, pp.337-351, (1972).

[46] Duffy, J. and Mindlin, R.D., Stress-Strain Relations and Vibrations of a Granular Medium, *J. Appl. Mech., (ASME)*, **24**, pp.585-593, (1957).

[47] Duncan, A.B., Peterson, G.P. and Fletcher, L.S., Effective Thermal Conductivity within Packed Beds of Spherical Particles, *J. of Heat Transfer*, **111**, pp. 830-836 (1989).

[48] Feda, J., Mechanics of Particulate Materials: The Principles, Developments in Geotechnical Engineering, **30**, Elsevier, (1982).

[49] Feng, S. and Sen, P., Percolation on Elastic Networks: New Exponent and Threshold, *Physical Review Letters*, **52**, No.3, pp.216-219, (1984).

[50] Finney, J.L, Random Packings and the Structure of Simple Fluids I. The Geometry of Random Close Packing, *Proc. Roy. Soc. London*, **319A**, pp. 479-493 (1970).

[51] Gilbert, E., *Theory of Shuffling*, Technical Memorandum, Bell Laboratories, 1955.

[52] Goddard, J.D., Huang, Y-H. and Huang, L-C., Rational Prediction of Composite Properties Based on Asymptotic Micromechanics, *Proc. 9th U.S. National Congress Appl. Mechanics*, pp. 197-211 (1982).

[53] Goddard, J.D., Dissipative Materials as Constitutive Models for Granular Materials, *Acta Mech.*, **63**, pp. 3-13 (1986).

[54] Goddard, J.D., Microstructural Origins of Continuum Stress Fields — A Brief History and Some Unresolved Issues, *Rec. Dev. in Structured Continua*, eds. D. De Kee and P.N. Kaloni, Longman Scientific & Technical/John Wiley (1986).

[55] Goddard, J.D., History Effects in Transient Diffusion through Heterogeneous Media, *Symposium Advances in Rate Processes*, Annual AICHE Meeting, Chicago, Nov. 11-16, 1990.

[56] Goddard, J.D. and Bashir, Y.M., On Reynolds Dilatancy, in *Recent Developments in Structured Continua*, ed. Kee, D.De and Kaloni, P.M., Vol. II, pp. 23-35, Longman Scientific and Technical/John Wiley, London (1990).

[57] Goddard, J.D., Nonlinear Elasticity and Pressure-Dependent Wave Speeds in Granular Media, *Proc. Roy. Soc. London*, **A430**, pp.105-131, (1990).

[58] Goddard, J.D., Zhuang, X. and Didwania, A.K., Microcell Methods and the Adjacency Matrix in the Simulation of the Mechanics of Granular Media, *Proc. of the 2nd International Conference on Discrete Element Methods*, eds. J.R. Williams and Graham G.W. Mustoe, pp.3-14, Intelligent Systems Engineering Laboratory, MIT, Cambridge, Massachusetts, (1993).

[59] Gourves, R., The analogical Bidimensional Model of Schneebeli. Applications to the Study of Micromechanics of Granular Media, *Advances in Micromechanics of Granular Materials*, eds. Shen, H.H. et al., pp. 353-362, Elsevier, Amsterdam (1992).

[60] Greenwood, J.A. and Williamson, J.B.P., Contact of Nominally flat surfaces, *Proc. Roy. Soc. London*, **A295**, pp.300-319, (1966).

[61] Greenwood, J.A. and Tripp, J.H., The Elastic Contact of Rough Spheres, *J. Appl. Mech.*, pp.153-159, March 1967.

[62] Harrigan, T. and Mann, R.W., Characterization of Microstructural Anisotropy in Orthotropic Materials using a Second Rank Tensor, *J. Mat. Sci.*, **19**, pp.761-, (1984).

[63] Head, K.H., Manual of Soil Laboratory Testing, Vol. **1**, **2**, and **3**, Halsted Press, (1980, 1982, 1986).

[64] Holm, R., Electric Contacts: Theory and Application, Springer-Verlag New York Inc., (1967).

[65] Horne, M.R., The Behavior of an Assembly of Rotund, Rigid, Cohesionless Particles. I & II, *Proc. Roy. Soc.*, **A286**, pp.62-97, (1965).

[66] Ishibashi, I., Chen, Y-C. and Jenkins, J.T., Dynamic Shear Modulus and Fabric: Part II, Stress Reversal, *Geotechnique*, **38**, pp.25-32, (1988).

[67] Jeffrey, D.J., Conduction through a Random Suspension of Spheres, *Proc. Roy. Soc.*, **A338**, pp.503, (1973).

[68] Jemaa, N.B., Queffelec, J.L. and Travers, D., Apparatus and Methods for Electrical Contact Resistance Study of Cleaned and Corroded Materials, *Measurement Science & Technology*, **1**, pp.282-286, (1990).

[69] Jenkins, J.T., Volume Change in Small Strain Axisymmetric Deformations of a Granular Material, *Micromechanics of Granular Materials*, eds. Satake, M. and Jenkins, J.T., pp. 143-152, Elsevier, Amsterdam (1987).

[70] Jenkins, J.T. and Strack, O.D., Mean-Field Stress-Strain Relations for Random Arrays of Identical Spheres in Triaxial Compression, *Advances in Micromechanics of Granular Materials*, eds. Shen, H.H. et al., pp. 41-50, Elsevier, Amsterdam (1992).

[71] Jodrey, W.S. and Tory, E.M., Computer Simulation of Isotropic, Homogeneous, Dense Random Packing of Equal Spheres, *Powder Tech.*, **30**, pp. 111-118 (1981).

[72] Johnson, K.L., Contact Mechanics, Cambridge University Press, (1985).

[73] Kanatani, K-I., Distribution of Directional Data and Fabric Tensors, *Int. J. Eng. Sci.*, **22**, No.2, pp.149-164 (1984).

[74] Keller, J.B., Conductivity of a Medium Containing a Dense Array of Perfectly Conducting Spheres or Cylinders or Nonconducting Cylinders, *J. Appl. Phys.*, **34**, pp.991, (1963).

[75] Kirkpatrick, S., Percolation and Conduction, *Reviews of Modern Physics*, **45**, No.4, pp.574-588, (1973).

[76] Kishino, Y., Discrete Model Analysis of Granular Media, *Micromechanics of Granular Materials*, eds. Satake, M. and Jenkins, J.T., pp. 143-152, Elsevier, Amsterdam (1987).

[77] Koenders, M.A. and Stefanovska, E., Simulation of an Assembly of Particles in Frictional Contact, *Proc. of the Third International Conference on Numerical Methods in Engineering: Theory and Applications*, Jan. 7-11, University College of Swansea, Swansea, Wales, U.K., pp.614-620.

[78] Konishi, J., Microscopic Model Studies on the Mechanical Behavior of Granular Materials, *Proc. of the U.S.-Japan seminar on Continuum-Mechanical and Statistical Approaches in the Mechanics of Granular Materials*, Tokyo, pp.27-45, (1978).

[79] Konishi, J., Oda, M. and Nemat-Nasser, S., Induced Anisotropy in Assemblies of Oval Cross-Sectional Rods in Biaxial Compression, *Mechanics of Granular Materials: New Models and Constitutive Relations*, eds. Jenkins, J.T. and Satake, M., pp. 31-39, Elsevier, Amsterdam (1983).

[80] Kytomaa, H., Liquefaction and Solidification, *Particulate Two-Phase Flow*, ed. M.C. Roco, pp.861-883, Butterworth-Heinemann, (1993).

[81] Lade, P.V. and Duncan, J.M., Cubical Triaxial tests on Cohesionless Soil, *J. Soil Mechanics and Foundation Div.*, , pp.793-812, Oct. (1973).

[82] Lambe, T.W. and Whitman, R.V., Soil Mechanics, John Wiley & Sons, Inc, (1969).

[83] Malucci, R.D., Multispot Model of Contacts Based on the Surface Features *Proc. of the Thirty-sixth IEEE Holm Conference on Electric Contacts and the Fifteenth International Conference on Electric Contacts*, Aug. 20-24, 1990, Montreal, Quebec, pp.625-634.

[84] Mindlin, R.D. and Deresiewicz, H., Elastic Spheres in Contact Under Varying Oblique Forces, *J. Appl. Mech.*, **20**, pp. 327-344, (1953).

[85] Mousseau, R.J. and Trump, R.P., Measurement of Electrical Anisotropy of Clay-Like Materials, *J. Appl. Phys.*, **67**, Oct. (1967).

[86] Nedderman, R.M., *Statics and Kinematics of Granular Materials*, Cambridge University Press, Cambridge, (1992).

[87] Nemat-Nasser, S. and Shokooh, A., A Unified Approach to Densification and Liquefaction of Cohesionless Sand in Cyclic Shearing, *Canadian Geotech. J.*, **16**, pp.659-678, (1979).

[88] Nemat-Nasser, S., On Behavior of Granular Materials in Simple Shear, *Soils and Foundations*, **20**, No.3, pp. 59-73 (1980).

[89] Nemat-Nasser, S. and Tobita, Y., Influence of Fabric on Liquefaction and Densification Potential of Cohesionless Sand, *Mechanics of Materials*, **1**, pp.43-62, (1982).

[90] Nemat-Nasser, S. and Mehrabadi, M. Stress and Fabric in Granular Masses, *Mechanics of Granular Materials: New Models and Constitutive Relations*, eds. Jenkins, J.T. and Satake, M., pp. 1-8, Elsevier, Amsterdam (1983).

[91] Nemat-Nasser, S. and Balendran, B., Micromechanics of Flow and Failure Modes of Particulate Media over a Wide Range of Strain Rates, *Advances in Micromechanics of Granular Materials*, eds. Shen, H.H. et al., pp. 21-30, Elsevier, Amsterdam (1992).

[92] Newland, P.L. and Allely, B.H., Volume Changes in Drained Triaxial Tests on Granular Materials, *Geotechnique*, **7**, pp.17-34, (1957).

[93] Ng, T.-T. and Lin X., Numerical Simulations of Naturally Deposited Granular Soil with Ellipsoidal Elements, *Proc. of the 2nd International Conference on Discrete Element Methods*, eds. J.R. Williams and Graham G.W. Mustoe, pp. 557-567 (1993).

[94] Oda, M., Initial Fabrics and Their Relations to Mechanical Properties of Granular material, *Soils and Foundations*, **12**, No.1, pp. 17-36 (1972).

[95] Oda, M., The Mechanism of Fabric Changes during Compressional Deformation of Sand, *Soils and Foundations*, **12**, No.2, pp. 1-18 (1972).

[96] Oda, M., Deformation Mechanism of Sand in Triaxial Compression Tests, *Soils and Foundations*, **12**, No.4, pp. 45-63 (1972).

[97] Oda, M., A Mechanical and Statistical Model of Granular Material, *Soils and Foundations*, **14**, No.1, pp. 13-27 (1974).

[98] Oda, M. and Konishi, J., Microscopic Deformation Mechanism of Granular Material in Simple Shear, *Soils and Foundations*, **14**, No.4, pp. 25-38 (1974).

[99] Oda, M., Significance of Fabric in Granular Mechanics, *Proc. of the U.S.-Japan seminar on Continuum-Mechanical and Statistical Approaches in the Mechanics of Granular Materials*, Tokyo, pp.7-26, (1978).

[100] Oda, M., Konishi, J and Nemat-Nasser. S., Some Experimentally Based Fundamental Results on the Mechanical Behavior of Granular Materials, *Geotechnique*, **30**, pp.479-, (1980).

[101] Oda, M., Nemat-Nasser, S. and Mehrabadi, M.M., A Statistical Study of Fabric in a Random Assembly of Spherical Granules, *Int'l J. for Numerical and Analytical Methods in Geomechanics*, **6**, pp. 77-94 (1982).

[102] Oda, M., Konishi, J. and Nemat-Nasser, S., Experimental Micromechanical Evaluation of the Strength of Granular Materials: Effect of Particle Rolling, *Mechanics of Granular Materials: New Models and Constitutive Relations*, eds. Jenkins, J.T. and Satake, M., pp. 137-149, Elsevier, Amsterdam (1983).

[103] Oda, M., Nemat-Nasser, S. and Konishi, J., Stress-induced Anisotropy in Granular Masses, *Soils and Foundations*, **25**, No.3, pp. 85-97 (1985).

[104] Oger, L., Bideau, D., Troadec, J.P. and Poirier, C., Effects of Disorder in the Behavior of the Schneebeli Models, *Advances in Micromechanics of Granular Materials*, eds. Shen, H.H. et al., pp. 363-371, Elsevier, Amsterdam (1992).

[105] Okada, N. and Nemat-Nasser, S., Energy Dissipation in Inelastic Flow of Saturated Cohesionless Granular Media, to appear in *Geotechnique*, (1994).

[106] Reiner, M., A Mathematical Theory of Dilatancy, *American J. Mathematics*, **67**, pp.350-362 (1945).

[107] Reynolds, O., On the Dilatancy of Media Composed of Rigid Particles in Contact with Experimental Illustrations, *Philos. Mag.*, *20*, pp. 469-481 (1885).

[108] Riordan, J., Review *Math. Rev.*, May 1950, 306 (1950).

[109] Rothenburg, L. and Selvadurai, A.P.S., Micromechanical Aspects of Plane Random Anisotropic Assemblies of Material Discs, *Proc. the Eighth Canadian Congress of Applied Mechanics*, pp.215-218, Moncton, June 7-12, (1981).

[110] Rothenburg, L. and Bathurst, R.J., Analytical Study of Induced Anisotropy in Idealized Granular Materials, *Geotechnique*, **39**, No.4, pp.601-614 (1989).

[111] Rothenburg, L. and Bathurst, R.J., Effects of Particle Shape on Micromechanical Behavior of Granular Materials, *Advances in Micromechanics of Granular Materials*, eds. Shen, H.H. et al., pp. 343-352, Elsevier, Amsterdam (1992).

[112] Rowe P.W., The Stress-dilatancy Relation for Static Equilibrium of an Assembly of Particle in Contact, *Proc. Roy. Soc.*, **A269**, pp. 500-527 (1962).

[113] Satake, M., Fabric Tensor in Granular Materials, *Deformation and Failure of Granular Materials*, eds. P.A. Vermeer and H.J. Lugar, Balkema, Rotterdam, pp.63-, (1982).

[114] Schreiner, W. and Kratky, K.W., Computer Simulation of Hard-Disc Packings with Spherical Boundary Conditions, *J. Chem. Soc., Faraday Trans. II*, **78**, pp. 379-389 (1982).

[115] Serrano, A.A. and Rodriguez-Ortiz, J.M., A Contribution to the Mechanics of Heterogeneous Granular Media, *Proc. Symp. on Plasticity and Soil Mechanics*, Cambridge (1973).

[116] Shante, K. S. and Kirkpatrick, S., An Introduction to Percolation Theory, *Advances in Physics*, **20**, pp.325-357, (1971).

[117] Shi, G., Discontinuous Deformation Analysis – A New Numerical Model for the Static and Dynamics of Deformable Block Structures, *1st U.S. Conference on Discrete Element Methods*, eds. G.G.W. Mustoe, M. Henriksen, and H-P Huttelmaier, (1989).

[118] Sidoroff, F., Cambou, B. and Mahboubi, A., Contact Force Distribution in Granular Media, *Advances in Micromechanics of Granular Materials*, eds. Shen, H.H. et al., pp. 31-40, Elsevier, Amsterdam (1992).

[119] Singer, M.T. and Kshonze, K., Electrical Resistance of Random Rough Contacting Surfaces Using Fractal Surface Modeling, *Proc. of the Thirty-seventh IEEE Holm Conference on Electric Contacts*, Oct. 6-9, 1991, Chicago, pp.73-82.

[120] Skinner, A.E., A Note on the Influence of Interparticle Friction on the Shearing Strength of a Random Assembly of Spherical particles, *Geotechnique*, **19**, pp. 150-157 (1969).

[121] Southwell, R.V., *Relaxation Methods in Engineering Science*, Oxford University Press, London, 1940.

[122] Spence, D.A., Self-Similar Solution to Adhesive Contact Problem with Incremental Loading, *Proc. Roy. Soc. London*, **A305**, pp.55-80, (1968).

[123] Subhash, G., Experimental Investigation of Fabric-Stress Relations in Granular Materials, M.S. thesis, University of California, San Diego, (1989).

[124] Subhash, G., Nemat-Nasser, S., Mehrabadi, M.M. and Shodja, H.M., Experimental Investigation of Fabric-stress Relations in Granular Materials, *Mech. Mater.*, **11**, pp. 87-106 (1991).

[125] Tamai, T., Electrical Conduction Mechanisms of Electric Contacts Covered with Contaminant Films, *Surface Contamination*, **2**, ed. K.L. Mittal, pp.967-982, Plenum, New York and London.

[126] Thornton, C. and Barnes, D.J., Computer simulated Deformation of Compact Granular Assemblies, *Acta Mech.*, **64**, pp.46-61 (1986).

[127] Throop, G.J. and Bearman, R.J., Numerical Solutions of the Percus-Yevick Equation for the Hard-Sphere Potential, *J. Chem. Phys.*, **42**, pp. 2408-2411 (1965).

[128] Ting, J.M., Corkum, B.T., Kauffman, C.R. and Greco, C., Discrete Numerical Model for Soil Mechanics, *J. Geotech. Eng.*, ASCE, **115**, pp. 379-397 (1989).

[129] Ting, J.M., Rowell, J.D. and Meachum, L., Influence of Particle Shape on the Strength of Ellipse-Shaped Granular Assemblages, *Proc. of the 2nd International Conference on Discrete Element Methods*, eds. J.R. Williams and Graham G.W. Mustoe, pp. 215-225 (1993).

[130] Turner, J.C.R., Two-Phase Conductivity: the Electrical Conductance of Liquid-Fluidized Beds of Spheres, *Chemical Engineering Science*, **31**, pp.487-492, (1976).

[131] Tutte, W.T., *Graph Theory*, Addison-Wesley Publishing Company, 1984.

[132] Uehara, Y., Ree, T. and Ree, F.H., Radial Distribution Function for Hard Disks from BGY2 Theory, *J. Chem. Phys.*, **70**, pp. 1876-1883 (1979).

[133] Vissher, W.M. and Bolsterli, M., Random Packing of Equal and Unequal in Two and Three Dimens'ons, *Nature*, **239**, pp. 504-507 (1972).

[134] Walton, K., The Effective Elastic Moduli of a Random Packing of Spheres, *J. Mech. and Phys. of Solids*, **35**, No.3, pp. 213-226 (1987).

[135] Walton, K., Numerical simulation of Inelastic, Frictional Particle-particle Interactions, *Particulate Two-Phase Flow*, ed. M.C. Roco, pp.884-911, Butterworth-Heinemann, (1993).

[136] Wood, G.C. and Melford, D.A., The Examination of Oxide Scales on Iron-Chromium Alloys by X-Ray Scanning MIcroanalysis, *J. Iron and Steel Inst.*, June 1961.

[137] Wood, W.W., *NPT*-Ensemble Monte Carlo Calculations for the Hard-Disk Fluids, *J. Chem. Phys.*, **52**, pp. 729-741 (1970).

[138] Yovanovich, M.M., Thermal Contact Resistance Across Elastically Deformed Spheres, *J. of Spacecrafts and Rockets*, **4**, pp. 119-122 (1967).

[139] Zhang, Y. and Cundall, P.A., Numerical simulation of Slow Deformation, *Proc. Tenth U.S. Congress Appl. Mech.*, Austin Texas, (1986).

[140] Zhuang, X., A Photochromic-tracer Method for Analysis of Strain Fields: — Application to 2D Granular Assemblages, M.S. thesis, University of Southern California, Los Angeles, California(1991).

[141] Zhuang, X., Computer Simulation and Experiments on the Quasi-static Mechanics and Transport Properties of Granular Materials, Ph.D. dissertation, University of California, San Diego, (1993).